DETECTING TEMPORAL ANOMALIES IN TIME SERIES DATA UTILIZING THE MATRIX PROFILE

Lappeenranta-Lahti University of Technology LUT

Mechatronic System Design Master's Thesis

2022

Alexander Beattie

Examiners: Dr. Pedro H. J. Nardelli

Dr. Heikki Handroos

ABSTRACT

Lappeenranta-Lahti University of Technology LUT

LUT School of Engineering Science

Mechatronic System Design

Alexander Beattie

DETECTING TEMPORAL ANOMALIES IN TIME SERIES DATA UTILIZING THE MATRIX PROFILE

Master's thesis

2022

78 pages, 16 figures, 7 tables

Examiners: Dr. Pedro H. J. Nardelli and Dr. Heikki Handroos

Keywords: Anomaly Detection, Time Series, Matrix Profile

This work presents a review of anomaly detection algorithms and libraries and the strengths and weaknesses of the most commonly used benchmarking datasets. With this information, the experimental datasets are selected and a practical implementation of an anomaly detector is created.

The matrix profile algorithm is selected for implementation because of its generalizeable approach for detecting real-time anomalies in streaming time series data. The STUMPY python library implementation of the iterative matrix profile is used for the creation of the detector. A series of custom filters is created and added to the detector to tune its sensitivity, recall, and detection accuracy.

Three experiments with significantly different conditions are presented to demonstrate the generalizability and performance of the detector. The experimential datasets used in this study include a hydraulic simulation, power electronic converter, and cyber-security intrusion detection dataset. With simple parameter tuning, the detector provides high accuracy and performance in a variety of difficult circumstances. In the future, the detector can be improved to further increase precision and accuracy in more complex circumstances. Additionally, the detector can be developed for use in a real-time, production system for monitoring and decision making.

ACKNOWLEDGEMENTS

My advisors, colleagues, and friends were instrumental in making this work possible. I would like to thank the following individuals and organizations for their contributions and support of this work:

- Dr. Pedro H. J. Nardelli for enabling growth and new research development through advisory and technical facilitation of this work.
- Dr. Heikki Handroos for review and oversight of this work.
- Pavol Mulinka for providing a technical overview of the FIREMAN project and implementation guidance for integrating this work with the FIREMAN project.
- Dr. Subham Sahoo for creating and sharing the PEC dataset and valuable insight and collaboration regarding the applicability and use cases for anomaly detection in power electronics.
- The FIREMAN research team for reviewing, collaborating, and supporting this work.
- Kate Highnam & Matti Bispham for discussing shortcomings and benefits of currently available research datasets in cyber-security.
- The Fulbright Finland Foundation and LUT University for a generous grant and stipend to complete a Master's in Mechatronic System Design at LUT University.
- CSC – IT Center for Science, Finland, for computational resources.

Table of Contents

List of Figures

List of Tables

1 Introduction

This work presents a review of anomaly detection algorithms, research datasets and an implementation of an anomaly detector that is robust to anomalies of different scales and characteristics. This section introduces the current knowledge gaps that prompted this work and the problems and research questions this work addresses. Additionally, an introduction to the FIREMAN project which accompanies the primary dataset are presented in this section.

1.1 Motivation

There is currently an implementation gap in the field of engineering between seminal theoretical research and applications in relevant domains. To implement new theoretical work, engineers must first study and understand relevant theoretical research. After understanding the research, developing and implementing a practical solution takes a considerable amount of time. This explains the long implementation delay between new scientific developments and industry usage.

This work presents a practical usage of a theoretical algorithm and demonstrates its applicability in three real-world datasets. Researchers, engineers, and industry professionals can utilize this work as a toolkit to develop new solutions without the time-consuming algorithm implementation process. This improves knowledge sharing and interdisciplinary collaboration.

The proposed toolkit enables accurate anomaly detection in systems from a wide variety of disciplines. This improves the sharing of knowledge and allows multiple domains to benefit from breakthroughs in specific areas. This yields many benefits including: (i) power savings from improved industrial control of processes; (ii) improved profit from better control over production output; and (iii) improved efficiency and accuracy in decision making processes.

Detecting anomalies in production systems is critical because current industrial control processes struggle with detecting and handling anomalies. A standard proportional–integral–derivative (PID) controller cannot determine or compensate for sensor failure or adversarial data. Additionally, applying machine learning algorithms to the domain of power systems and industrial processes is rapidly gaining interest from stakeholders.

In this domain, it is critical to provide algorithmic explainability to confirm algorithms react in a predictable way to adversarial inputs. If an algorithm produces an unpredictable output it could create an attack vector. If a unpredictable input or cyber-attack exploits this it can lead to significant safety hazards and financial implications.

1.2 The FIREMAN Project

Over the course of 3 years, 6 partner universities are developing a "**F**ramework for the **I**dentification of **R**are **E**vents Via **Ma**chine Learning and IoT **N**etworks known as the FIREMAN project" (Fireman Project EU, 2021). FIREMAN is a multidisciplinary cooperation between 6 universities in 4 different countries. Lappeenranta–Lahti University of Technology (LUT) serves as the project coordinator and is involved in all aspects of the project.

The project is partitioned into overall Work Packages (WPs) that define overall sub-tasks to meet the general project objective. Although there are many research components of the FIREMAN project which are described in detail in Section 2.1, the primary focus of this work is on the anomaly detection. This work provides both theoretical and concrete approaches for anomaly detection in streaming time-series data.

The project goals include: (i) improving interdisciplinary collaboration to create end-to-end cyber-physical systems solutions; (ii) creating a framework that integrates the entire cyber-physical ecosystem from remote sensing and data acquisition to analysis and decision making; and (iii) detecting, processing, and handling anomalies in a diverse set of environments and application areas.

It is essential to ensure that all stakeholders remain informed throughout the project. Conveying complex information to stakeholders from a variety of disciplines is challenging. This work demonstrates the efficacy and viability of a component of the FIREMAN project and contributes to solving this problem. This creates a value proposition for the diverse group of project stakeholders.

1.2.1 Power Electronics Converter Collaboration

Aalborg University (AAU) and LUT University have collaborated to implement and test the theoretical work developed in FIREMAN on a real-world problem. AAU has a Power Electronics Converter simulation environment where different anomalies and perturbations can be introduced into a real-world power system. The inputs and outputs of the system are recorded and a collection of these trials has been used to create the Power Electronic Converter (PEC) Dataset referenced in this work.

This work creates the foundation for the implementation of a production, real-time anomaly detection solution. The preliminary anomaly detector and results on the PEC dataset serve as a feasibility study for further implementation. In the future, this work can be expanded to a production solution for real power electronics systems. Many of the research problems and questions posed below were formulated through this FIREMAN collaboration.

1.3 Research Problem

There are many existing techniques for outlier detection in a variety of disciplines from batch machine learning to conventional statistical approaches. These techniques are generally incompatible and siloed. This research proposes to break that barrier and utilize the best techniques and approaches for the problem.

An emerging area of research is applying machine learning techniques to data streams. Streaming or 'online' machine learning algorithms are unable to look at the data multiple times and must act on and update the model as new datapoints arrive. There is a research gap in implementing and testing these algorithms against existing methods for stream anomaly detection (ex. Half-Space Trees) in popular libraries as many of them are two or three years behind current breakthroughs.

These techniques for anomaly detection and machine learning in general are also new in the field of power electronics. Using a machine learning pipeline to improve over an existing static controlled system presents many opportunities. Many machine learning algorithms a black-box system where the reasons for the output classifications are unknown. This presents challenges for system critical applications, like power systems, where it is essential to understand why an algorithm is making a specific decision.

In this work, algorithm explainability is analyzed and methods are introduced that explain why the algorithm is making certain decisions based on input data. This is used to analyze the impact of adversarial data on the system in the context of cyber-security to design a controller that can defend against potential threats from adversarial inputs. The developed detector provides an explainable model that can be used to control industrial processes.

1.4 Research Questions

In order to improve the relevance and usability of this study, a practical application of anomaly detection is investigated in relation to the field of power electronics. Authors Sahoo, H. Wang, and Blaabjerg, 2021 propose two fundamental questions for the field of power electronics: (i) How can researchers ensure *trust* and *confidence* in the output of machine learning algorithms in power electronics? (ii) How does the physical power electronics system correspond to the output of machine learning algorithms?

With the research problems presented in Section 1.3 and the power electronics questions above, the research questions are formulated as follows:

- What are the different anomaly classification types?
- What datasets are best for testing and benchmarking anomaly detection algorithms?
- How can algorithmic techniques be used to enable increased explainability, performance, and security in system critical applications?
- What existing machine learning libraries are available for streaming time series analysis and how can they be implemented into a detection strategy?

2 Background

This section presents a review of existing algorithms, an anomaly taxonomy, and an evaluation of existing anomaly detection libraries. Additionally, this section includes an overview of the Work Packages (WPs) that define the deliverables of the FIREMAN project.

2.1 FIREMAN Work Packages

This subsection outlines the progress and overall goal of the milestones or work packages that guide the FIREMAN project. The interconnection and overview of the individual WPs is shown in Figure 1

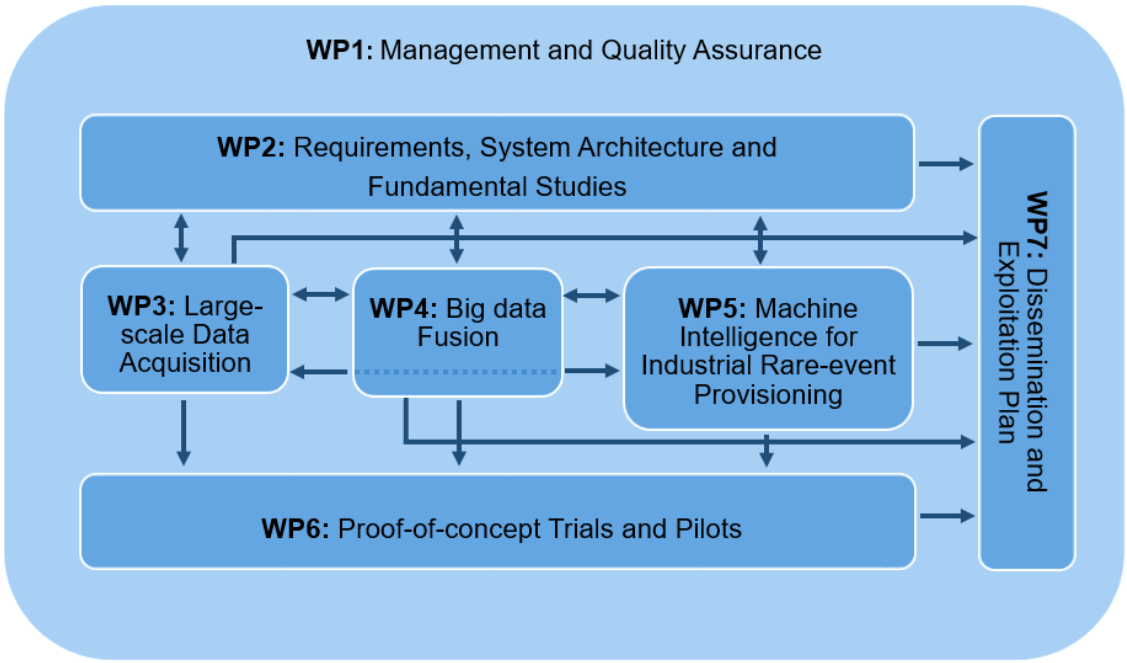


Figure 1. FIREMAN WP Diagram (Fireman Project EU, 2021)

This work focuses on WP6 discussed in Section 2.1.4 by demonstrating an implementation of a theoretical algorithm for real-world anomaly detection.

2.1.1 Large Scale Data Acquisition (WP3)

This work package focuses on "event-based modelling and traffic characterization techniques aiming for a reduced use of communication and storage resources. This pre-processing task is expected -among others- to optimize the subsequent data transmissions by injecting only relevant data in the network to reduce overhead and increase spectral efficiency." (Rojas, Nardelli, Kalalas, and Papadias, 2020b.)

In this portion of FIREMAN, we examined a variety of industrial datasets including the Tennessee Eastman Process, Electricity Metering, IEC-61850 Distribution Automation, and the EPFL Smart Grid Pilot. This dataset is not heavily reliant on relational constraints or atomicity, consistency, isolation, durability (ACID) compliance. Because of this, we have selected a non-relational PostgreSQL database for storage because of its superior throughput and increased performance in data processing applications.

The sensors traditionally used in Cyber-Physical Systems do not have much compute or storage capacity which makes pre-processing of data at the sensor level challenging. Because of this, "compression is essential for reducing the challenge of data storage, collection, transmission, processing, and analysis at the local level"(S. Li, L.D. Xu, and X. Wang, 2013). Authors Rojas, Nardelli, Kalalas, I. Christou, et al., 2020a have shown that it is possible to achieve a 92.6% compression rate which means that only 7.4% of samples are transmitted. We have also used different methods including linear interpolation to de-compress the data and have measured the error using the root mean square technique.

Modeling of transmission reduction techniques are centered around the electricity modeling dataset (Castro Tomé et al., 2022). Three approaches for reducing transmissions were presented including transmissions on large spikes, transmissions on accumulated variance, and time interval defined transmissions with a timeout to

compensate for meter or network failure. These simulations have shown a large potential for reducing transmissions without significantly affecting the accuracy of the measured signal. We modeled a Markov-Modulated Poission Process to simulate these industrial processes for use in further stages of the project.

2.1.2 Big Data Fusion (WP4)

The goal of this task is "to show how a large number of sensors and their corresponding data can be accommodated and aggregated at small cost for reliably detecting rare events"(Souza Sant'Ana et al., 2020). This task focuses on three primary objects: (i) heterogeneous data aggregation in machine-type communications; (ii) signature-based cluster formation; and (iii) IoT platform and database selection.

Data aggregation is important because it is challenging to simultaneously connect a massive number of devices. This strategy outlined by Authors Dawy et al., 2017 relies on the following principals: (i) decreasing communication distance and power consumption of connected devices; (ii) utilizing an efficient distributed node routing network to decrease bandwidth congestion of central nodes; and (iii) extending the network coverage. The researchers also introduced a cluster formation scheme based on signatures that reduces the signalling overhead required for peer-discovery in the network. This technique utilizes signal aggregates to reduce traffic congestion to the central node.

Additionally, we surveyed IoT experts to determine the most important properties and components of an IoT system. This survey compared the five most popular cloud IoT providers: AWS, Azure, GCP, IBM Watson, and Oracle IoT to create a comparison that can be used by businesses when selecting a cloud provider that suits their business needs (Ullah et al., 2020).

SEAT, an automotive component manufacturer, provided two use cases and accompanying datasets for the FIREMAN project. The first involves early failure detection of mechanical components in the drive chain in the Paint shop which causes axial displacement. The second involves detecting early failure of the spindle on a CNC machine which ensures lineal movement over a surface. In both of these situations, detecting failure early helps SEAT solve problems before they begin to impact production components. This can eventually reduce production downtime to zero.

2.1.3 Machine Intelligence for Industrial Rare-event Provisioning (WP5)

The goal of this task is to utilize machine intelligence techniques to predict indicators of health for a machine, component, or entire industrial process to detect premature failure. The main technique proposed in this task is the Quantitative Association Rule Mining Algorithm (QARMA).

"QARMA is a family of algorithms for extracting all (or, depending on user inputs, an important subset of) valid non-dominated quantitative association rules that hold in a dataset, that can then be used for further data analysis such as deriving rule-based classifier ensembles or as explainers of classification results of other black-box classifiers."(I.T. Christou et al., 2020.) This technique is one of the most human understandable techniques for machine intelligence and has shown promise for this application. In this WP, QARMA is tested on the SEAT dataset discussed in Section 2.1.2.

Traditional methods for pruning the rules created by the algorithm were analyzed in this WP as many of them can be extraneous or unreliable. The fastest open source Mixed-Integer Programming (MIP) tool was able to solve this problem in 19 seconds while the parallelized hybrid search algorithm proposed in this WP solved the problem 240 times faster when using parallel threads.

Authors I.T. Christou et al., 2020 used a breadth first search approach to compute a 'small' subset of the rules extracted by QARMA that cover the majority of instances in a training dataset. Through experimentation on a synthetic power grid fault diagnostic dataset, QARMA performs better than certain neural networks with noisy data (Gutierrez-Rojas et al., 2022). This is because of the over-fitting of training data with Deep Neural Network architectures. The rule based methods generated by QARMA produce human understandable rules which is beneficial in creating explainable AI.

2.1.4 Proof-of-Concept Trails and Pilots (WP6)

The primary objectives of this task as stated by Authors (Souza Sant'Ana et al., 2021) using FIREMAN approaches are: (i) testing and deployment of theoretical techniques presented thus far; (ii) specific proof-of-concept demonstrations and simulations ; and (iii) implementation plans for production, real-time monitoring.

Authors (Alves et al., 2022) describe the Power Electronic Converter (PEC) setup at Aalborg University (AAU) that is discussed in this work. The test setup facilities studying microgrids and the impact of disturbances or faults on power transmission and reliability. With the test setup, it is possible to perturb system parameters and measure their impact on the overall system with various sensor measurements. These simulation conditions are used to create the PEC dataset discussed in Section 3.4.2.

2.2 Algorithm Explainability

Certain high-risk use cases for machine learning algorithms demand a high level of explainability and confidence in the algorithm outputs to use in decision making. In many cases an algorithm makes a classification decision but there is not a clear explanation as to why. In the field of power electronics, understanding why a data-driven controller is making a decision is critical to incorporating it into real world power distribution scenarios.

2.2.1 Types of Uncertainty

Determining sources of uncertainty in a model or system is critical. When analyzing uncertainty, it is important to understand the two different types: (i) epistemic and (ii) aleatoric uncertainty.

Epistemic uncertainty results from inadequate training data. In this case, the training data is not sufficient to provide enough or accurate data to the model. This can result from imbalanced or insufficient training data. Increasing the amount of training data or decreasing class imbalance can help reduce this type of uncertainty.

Aleatoric uncertainty arise from probabilistic errors in sampling that follow a specific probability distribution. This type of uncertainty is independent to the amount of data collected and therefore cannot be corrected with more training data. If a signal has noise inline with a given probability distribution, having more data on that signal does not change the noise probability distribution. This type of uncertainty references the distribution of random errors in the data and not the data distribution itself.

For illustration, suppose there is a continuous audio recording at a train station. There are three announcements each hour that disrupt the recording but their occurrence in the hour is unknown. Having more data (hours), does not change the amount of announcements that occur. In the systems domain, a malfunctioning sensor or bad connection can generate noise (a type of aleatoric uncertainty) in a similar way that cannot be fixed by more measurements or more data.

Data imputation can be used as a technique to correct for or understand this type of uncertainty. If you can identify that the uncertainty is present, then you can use data imputation to form a well educated guess as to what the value should actually be. The significance of identifying aleatoric uncertainty is that no amount of model tuning or data collection can fix the underlying issue, so data imputation presents an opportunity that standard techniques do not.

2.3 Combating Model Uncertainty

With deep learning algorithms, it is important to know the confidence level of the model output. Authors Goodfellow, Shlens, and Szegedy, 2014 explain that adding simple adversarial data (like small noise to a photo) makes an image recognition algorithm incorrectly classify one animal as completely unrelated one. This is further concerning since adversarial data does not need to be tailored to a specific algorithm. The transferability of this type of adversarial data allows it to be applied to many black box algorithms to achieve an unintended or potentially malicious result.

To solve this problem, Authors Sahoo, H. Wang, and Blaabjerg, 2021 propose using conditional entropy to determine how each input is related to each output. The generated plots are then compared to the physical insights of the system. In the process, adversarial data that falls outside the plot is identified and removed, and the model is retrained. This technique is helpful for identifying and removing adversarial data that falls outside the range of accepted values. Unfortunately, it does not identify data overloading in a specific portion of the graph which would create an incorrect classification.

2.3.1 Bayesian Dropout

Bayesian techniques can be used to create a probability distribution over the weights of each neuron to determine a level of prediction uncertainty. Authors Blundell et al., 2015 introduce a standard Bayesian neural network implementation with back-propagation that can determine these probability distributions.

This method is effective at improving model accuracy but Authors Gal and Ghahramani, 2016 explain that retraining a large number of models on a variety of datasets is computationally expensive and time consuming. A dropout technique can be used to approximate the Bayesian representation with improved computational efficiently. Authors Gal and Ghahramani, 2016 explain that this technique avoids over-fitting by randomly sampling and dropping network nodes across many different training iterations.

Performing this dropout technique while training and testing the algorithm enables the computation of variance to determine the uncertainty level of the outputs. This enables researchers to determine if an algorithm is providing a best guess answer with high levels of uncertainty for specific values. This can signal the need for human intervention or review before making decisions based on the algorithm output.

2.3.2 Shapely Additive Explanations (SHAP)

Using reverse-engineering, the output of a machine learning algorithm can be analyzed and explained. Authors Lundberg and Lee, 2017 introduce **SH**apely **A**dditive ex**P**lanations (SHAP) to interpret and explain the *why* behind machine learning algorithm results.

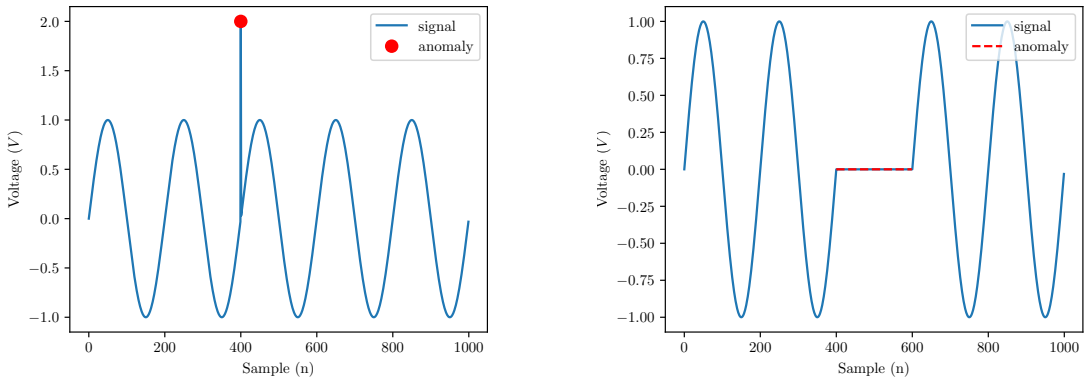
Machine learning models usually output the likelihood of a certain prediction given a set of inputs. SHAP explains why the model made that classification decision. To determine this, SHAP analyzes every possible combination of input weights to determine how significantly each input contributes to the overall output.

While this technique yields accurate explanations of black-box machine learning outputs, it is still a reverse-engineering technique. It cannot explain a model that is not yet trained with unknown outputs. This is problematic for online techniques where the model is created and retrained as new data inputs enter the pipeline. As the model is updated, the decision making parameters that affect the output change, and the output looses explainability.

This demonstrates the importance of explainability in techniques that do not have traditional machine learning models. In this case, it is important to have an explainable set of rules or behaviors that govern the model before inference. Reverse-engineering is not effective for this use case since the inputs are constantly updating the model. Explanations from reverse-engineering would only be valid for the present iteration and not generalizeable to subsequent predictions.

2.4 Outlier Taxonomy

The term 'outlier' or 'anomaly' can have a variety of meanings depending on the context. In order to select appropriate techniques for outlier detection, it is essential to create a taxonomy for various outlier types. Outliers are primarily categorized as: (i) point-wise and (ii) context-wise. These distinction between these two types is illustrated graphically in Figure 2.



(a) Point-wise Outlier (b) Context-wise Outlier

Figure 2. Outlier Taxonomy (Adapted from Lai, Zha, J. Xu, et al., 2021b)

2.4.1 Point-wise Outliers

Point-wise outliers are single points that are anomalous in respect to the global dataset. This can cause many problems in machine learning algorithms, and a large body of outlier detection research is focused around this area. These outliers can be non-temporal or temporal in nature and often represent phenomenon like intermittent sensor failure.

Additionally, these types of outliers can skew scaling and normalization operations. It is important to consider the presence of these types of outliers when selecting which type of scaler to use. Minimum-Maximum scaling is a popular choice for scaling a dataset but it is not robust to outliers.

2.4.2 Context-wise Outliers

Context-wise outliers represent a series of points that are anomalous. They can be further described based on their reference frame: (i) local and (ii) global.

Local contextual outliers are anomalous in respect to a specific sub-set or window of the dataset. Global contextual outliers are similar to local contextual outliers but the window size is the global dataset. Singular points within the global contextual outlier sub-set are usually not anomalous but the entire pattern is.

Generally these two outlier types are treated as pattern-wise contextual outliers with differing window sizes. These outliers occur often in time-series data because of the interdependence of samples and sampling time.

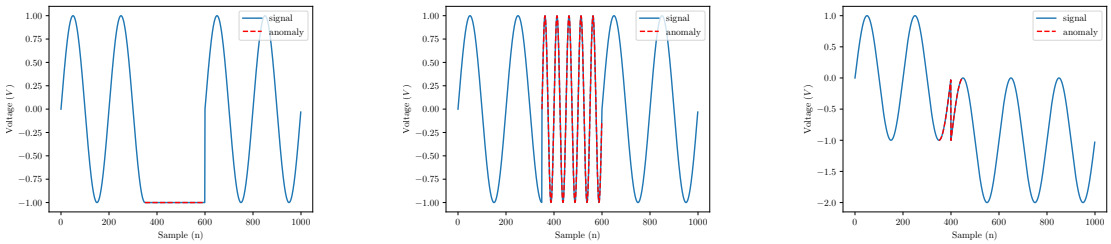
Figure 2b shows that points in a pattern-wise outlier represent a phenomenon that is anomalous with respect to a reference frame. Identifying the anomaly accurately requires selecting the size of the reference frame carefully. This study analyzes the detection challenges presented by context-wise outliers.

2.5 Context-wise Outlier Sub-Categories

Since context-wise outliers are the focus of this study, it is possible to further classify this outlier type into three sub-categories: (i) shaplet (ii) seasonal and (iii) trend outliers. This subset of anomalies represents various anomalous sub-sequences of the data in a given context that are described in detail in Table 1. Figure 3 represents these phenomenon graphically.

Table 1. Context-wise Anomaly Classifications

Class	**Description**
Shaplet	Shaplet outliers are classified by a shaplet or pattern that significantly differs from the normal data pattern. This outlier type classifies abrupt faults in a system and is an important outlier type for this study.
Seasonal	Seasonal outliers are classified by an increased or decrease pattern frequency during a specific time period. Identifying seasonal outliers is important in understanding specific phenomenon like a spike in web traffic related to a major holiday. Another example is an increased demand in residential electrical demand because of a large televised sporting event.
Trend	Trend outliers are classified by a sub-sequence of the dataset that modifies the underlying distribution of the data. Trend outliers are present in certain faults in the PEC dataset and in certain attacks in the BETH dataset. This is discussed further in Section 3.4.



(a) Shaplet Outlier (b) Seasonal Outlier (c) Trend Outlier

Figure 3. Context-wise Outlier Taxonomy (Adapted from Lai, Zha, J. Xu, et al., 2021b)

2.6 Anomaly Detection Algorithms

Existing outlier detection techniques are traditionally applied to a singular domain specific problem. Many techniques are versatile (especially unsupervised ones) and can be applied in a much more general context. This section focuses on current developments and advances across all disciplines in anomaly detection techniques.

While there are many new and state-of-the-art developments, corresponding code to a publication is often not included. This makes reproducing the results presented in most papers difficult. In order to utilize these techniques in practice, a time-consuming algorithm implementation process based on pseudo-code is required. Section 2.7 examines existing implementations of some of the common existing techniques.

2.6.1 Streaming Half Space Trees (HST)

Authors Tan, Ting, and Liu, 2011 introduce Streaming Half Space Trees (HST) as a single-class detection technique for changing data streams. It is an efficient method for point-wise outlier detection and is implemented in a variety of anomaly detection libraries. This technique relies on ensembleing HS-Trees which are full binary trees, where all leaves are at the same depth. It is fast and efficient with a time and space complexity of $O(1)$, and is suitable for handling streaming data.

2.6.2 Local Outlier Factor (LOF)

The Local Outlier Factor (LOF) technique assigns a degree of 'outlierness' to each datapoint as opposed to a binary classifier. The traditional LOF method is used to detect outliers in static datasets. It must compute neighborhood distances for each data point. These distances must be recomputed entirely when a new datapoint is added so it is not well suited for streaming data. Many researchers including Authors Na, D. Kim, and H. Yu, 2018 and Salehi et al., 2017 have proposed streaming adaptations to the base LOF algorithm. These extensions modify the algorithm to be suitable for stream learning by improving speed, memory usage, and computational efficiency. This algorithm is well suited to point-wise contextual outlier detection.

2.6.3 Matrix Profile

The Matrix Profile algorithm computes the distances between neighboring points in a dataset. The matrix profile technique has a variety of advantages including speed and generalizability to a variety of problem domains.

Authors Yeh et al., 2016 explain that in the algorithm, the results for each point or sub-sequence are stored in a vector and the combination of all the sub-sequences forms the overall matrix. The conventional algorithm utilizes the Euclidean method to determine point-wise distance but other distance calculations can also be utilized. Z-normalization is traditionally used to scale the data with this technique. It is not required though and the normalization technique can be modified or omitted depending on the specific dataset requirements.

Using a fixed (m) sized window, the algorithm computes the nearest-neighbor distances compared to the entire data stream. The results of the computation and results of the index of the closest neighbors are stored in order of closeness in a new index. Author Marrs, 2019 outlines the overall algorithm procedure as follows:

1. For each point in the window (m), compute the distance to the nearest neighbor against the entire data set.

2. Exclude identical or nearly identical matches to prevent inaccuracy.

3. Update the distance matrix with the new closest neighbor distance.

4. Set the position matrix with the index position of the new closest neighbor.

Using this technique, it is easy to extract information, like motifs or repeated patterns, from the dataset. More importantly, discords which represent anomalies can be discovered from a data stream with the Matrix Profile technique.

2.7 Anomaly Detection Libraries

Libraries and algorithms are surveyed from a variety of Github repositories, online sources, and a comprehensive framework list from Author Medico, 2020. The most relevant libraries are selected and their performance is evaluated against the sample datasets in Section 4. The following criteria are required for a framework to be selected for analysis: (i) written for Python (ii) compatible with MacOS, Windows, and Linux operating systems (iii) actively maintained with git commits within the past two years (iv) suitable for context-wise anomaly detection on signals (v) quality documentation (vi) open-source with over 100 stars.

Some libraries were excluded from selection because their focus on production environments makes them difficult to test and perform research with. In further research, these libraries may be considered to implement a production outlier detection pipeline. Many of them use the underlying algorithms described in Section 2.6 for detection.

Table 2 compares the selected outlier detection libraries in a systematic way. This table can be used to quickly compare and evaluate libraries for use in implementation. The specific algorithms for each library are classified into a category with a short description.

Each method is classified according to how suitable it is for online learning and its primary use case. For the algorithm to have a yes in the online learning category, the algorithm must be able to iteratively update the model efficiently, and predict specific instances one at a time. Partially online means that there is a pre-trained model, with individual instance detection. Not online means that the algorithm is not suited for online detection.

Table 2. Outlier Detection Overview [(**PA**): Point-wise Anomaly, (**CA**): Context-wise Anomaly, (**DD**): Drift Detection, (**S**): Segmentation)]

Library	**Algorithm**	**Focus**	**Online**	**Description**
PySAD[1]	Indivdidual Models	**PA**	yes	Score anomalies
	Probability Calibrators	**S**	yes	Convert raw scores to probabilities
PyOD[2]	Individual Models	**PA**	no	Score anomalies
River[3]	Individual Models	**PA**	yes	Score anomalies
	Drift models	**DD**	yes	Detect concept drift
	Time-series models	**S**	yes	Segment time-series data

Continued on next page

[1]Yilmaz and Kozat, 2020.
[2]Zhao, Nasrullah, and Z. Li, 2019.
[3]Montiel et al., 2020.

Table 2 – *Continued from previous page*

Library	**Algorithm**	**Focus**	**Online**	**Description**
STUMPY[4]	Matrix Profile[5]	**CA**	yes	Time-series subsequence analysis
	Time Series Chains[6]	**CA, DD**	no	Motif drift detection
	FLUSS & FLOSS[7]	**CA, S**	yes	Time-series segmentation
tsod[8]	Simple Detectors	**CA, PA**	partial	Gradient and range detectors
TODS[9]	Detection Algorithms	**CA, PA**	partial	Detect anomalies
	Feature Analysis	**S**	partial	Statistics based segmentation
Alibi Detect[10]	Outlier Detection	**CA, PA**	partial	Anomaly detection
	Adversarial Detection	**CA**	partial	Detect adversarial data
	Drift Detection	**DD**	partial	Concept drift identifications
banpei[11]	Hotelling's Theory	**PA**	yes	Outlier detection
	SST	**S**	yes	Change point detection

Continued on next page

[4]Law, 2019.
[5]Yeh et al., 2016.
[6]Zhu et al., 2017.
[7]Gharghabi et al., 2017.
[8]Andersson, Mariegaard, and Falk, 2021.
[9]Lai, Zha, G. Wang, et al., 2021a.
[10]Van Looveren et al., 2019.
[11]Tsuruta and Feng, 2021.

Table 2 – *Continued from previous page*

Library	**Algorithm**	**Focus**	**Online**	**Description**
SaxPy[12]	SAX	**PA, S**	no	Symbolic Aggregate approXimation
	EMMA	**CA, S**	no	Time-series motif discovery
Darts[13]	Filtering Models	**CA**	no	Gaussian, Kalman, Moving Average Filter
	Forecasting Models	**S**	no	Predictive forecasting
Merlion[14]	Anomaly Algorithms	**CA, PA**	partial	Anomaly Detection strategies
	Forecasting	**S**	partial	Predictive forecasting
Luminaire[15]	Structural Modeling	**PA**	yes	Anomaly detection
	Windowed Density Model	**CA, DD**	yes	Anomalous window detection using KL divergence

2.8 Library Details

The libraries from Table 2 are presented and analyzed for suitability in this section. The analysis is performed with information provided by the libraries, personal experience, and the results from Table 2.

[12]Senin et al., 2018.
[13]Herzen et al., 2021.
[14]Bhatnagar et al., 2021.
[15]Chakraborty et al., 2020.

PySAD (Yilmaz and Kozat, 2020) can operate in a streaming context so models can be updated as new data points arrive. This is both computational and memory efficient. PySAD also provides a variety of pre-processing, post-processing, and probability calculation tools. This library provides both univariate and multivariate prediction models for supervised and unsupervised data. Its focus is on point-wise anomaly detection and segmentation which makes unsuitable for use in this study.

PyOD (Zhao, Nasrullah, and Z. Li, 2019) is designed for offline data which makes it unsuitable for this study. It includes a vast collection of classic and modern anomaly detection algorithms, some of which could be adapted to run in a streaming context. There are several performance optimizations present in the library, like multi-core parallelism, which could also be utilized. The repository has over 5,000 stars on GitHub and is a very popular python framework for anomaly detection.

River (Montiel et al., 2020) is a streaming machine learning library that provides tools for anomaly detection, drift detection, and more. It is the result of a combination of two python libraries: (i) creme (ii) and scikit-multiflow. It is a fast and efficient library that can learn and predict with single instances. With over 3,000 start on GitHub it is a popular library. It includes only one anomaly detection model which is focused on point-wise anomaly detection that is not suitable for this study.

STUMPY (Law, 2019) is a library used to compute the uni-variate or multi-variate matrix profile for provided time series data created by U.S. based investment firm TD Ameritrade. In its simplest form, the algorithm compares element-wise distances between each sub-sequence for each point. This is known as a self-similarity join. This naive method which is defined in Section 2.6.3 is very computationally expensive with a complexity of $O(n^2m)$. STUMPY introduces a computationally efficient method with gpu acceleration to improve the matrix profile computational performance.

With over 2,000 stars on GitHub the libaray is rapidly gaining popularity. It is versatile for a variety of anomaly detection tasks and can operate in an online or real-time context. STUMPY is a good choice for this study because of its focus on contextual anomalies and ability to run incrementally in resource constrained environments. Specifically it is well suited to analyzing the contextual shaplet outliers present in the datasets and finding anomalies, called discords. This library is used to create and analyze a uni-variate Matrix Profile detector for the experimental datasets.

Tsod (Andersson, Mariegaard, and Falk, 2021) is a library focused on anomaly detection in relation to water. Although it is designed for the water domain, the techniques can be analyzed and used in a variety of other domains. It includes a gradient detection technique for detecting abrupt changes that was ultimately unsuccessful for detecting the contextual shaplet outliers. Tsod includes a contextual anomaly detection method but is only partially online. This library is experimentally tested in this study.

TODS (Lai, Zha, G. Wang, et al., 2021a) provides techniques for partially online, multi-variate anomaly detection. It is still gaining popularity with slightly more than 500 stars. Although it includes many modern techniques and optimizations, it requires training models and creating complex pipelines. Because of this, TODS is not an ideal choice for this study.

Alibi Detect (Van Looveren et al., 2019) is a library developed by U.K. based Seldon for outlier, adversarial, and drift data detection compatible with TensorFlow and PyTorch backends. There are a variety of supported algorithms for these three types of data detection tasks and the package is actively maintained. The company also provide and maintain an enterprise machine learning deployment platform that integrates with Alibi Detect. The framework handles streaming and offline data detection for time series, tabular, text, and image data. This libraries focus on production implementation makes it unsuitable for use in this work.

Banpei (Tsuruta and Feng, 2021) is focused on point-wise anomaly detection and segmentation. It implements the Singular spectrum transformation (SST) algorithm for change point detection and Hotelling's theory for outlier detection. It was designed to offer real time monitoring functionality and operates on streaming data. Because it does not offer contextual anomaly detection, it is not a good candidate for the datasets presented in this study.

SaxPy (Senin et al., 2018) uses Symbolic Aggregate approximation (SAX) to transform a series of numerical data points into a sequence of letters. The library implements HOT-SAX, a time series anomaly detection algorithm. Unfortunately, the library does not operate online and is not suitable for this study.

Darts (Herzen et al., 2021) is python machine learning package developed by the Swiss AI company Unit8 for time series data forecasting. The library provides a variety of advanced and classic models with a sci-kit learn compatible interface. It boasts a wide array of features and advanced techniques but does not run in an online context and is not suitable for this study.

Merlion (Bhatnagar et al., 2021) is a library developed by the U.S. company SalesForce. It includes many available datasets and techniques for experimentation as well as a full tool chain to benchmark and interpret results. It requires model training and includes an automatic hyper-parameter tuner in the library. Because the library requires prior training of a model and does not operate in an online context, it is not suitable for this study. In the future, some of the techniques and pipelines from the library could be adapted to a streaming context.

Luminaire (Chakraborty et al., 2020) is a library developed by the U.S. real estate company Zillow. It includes many anomaly detection features for point-wise and contextual outliers. This library provides a technique to monitor data points over a window of time which is helpful in detecting context-wise outliers. Similar to the Merlion library, Luminaire requires prior training of models but can run inference in a streaming environment which makes it unsuitable for this study.

2.9 Summary

This section presents the libraries, techniques, and methods available for outlier detection. Additionally, the FIREMAN project work packages are summarized and presented in Section 2.1. This work contributes to WP6 in Section 2.1.4 by creating an anomaly detection system. The next section discusses the specific methods used in this work to create a generalizeable anomaly detector.

3 Methods

In this section, a literature review is conducted to understand and evaluate effective algorithms and techniques for outlier detection. This review includes scientific literature, library documentation, and online resources. The goal of this review is to understand the cutting edge techniques and the logic behind algorithm decision-making. Additionally, a variety of interdisciplinary methods (*e.g.*, Statistics, Deep Learning, etc.) are used to compare ideal use cases for building an anomaly detection pipeline and toolkit.

A review is also presented that determines the most common datasets used to benchmark anomaly detection techniques. Since anomalies are by nature rare, most datasets contain a very small number of them. It is critical that the anomalies they do contain are a good representative sample. Results in Section 3.3 show the most common datasets currently used in literature.

Data collection in this study is performed using the following procedure:

1. Pre-process data inline with recommendations from the dataset authors.

2. Setup the data processing pipelines using the developed anomaly detector.

3. Execute the pipeline for each experimental dataset.

4. Tune the detector parameters and optimize for the specific dataset.

5. Collect and graph the results for anomaly detection analysis.

Data analysis is performed by comparing the results of the detector against the ground truth for the experimental datasets. Through this analysis the benefits and shortcomings of each method can be understood. This analysis also helps determine what type of detection methodology works best for the anomalies in this study. Using this analysis, the best method can be implemented for solving real-world interdisciplinary problems.

3.1 Measuring Algorithms and Methods

Algorithms are susceptible to missing detections from challenging phenomena or false positives from non-anomalous points. A successful detector must weight false detections lower than missed detections. If a detection is missed, it is much more significant than a false positive, because a false positive can be ignored but a missed detection cannot. Because of this, many existing algorithms and implementations are being compared to identify the best for the contextual outliers present in the experimental datasets. This provides insight on the existing shortcomings in the field and shows where the algorithms can be improved.

There are existing industry standards for comparing machine learning techniques. Some of these include ROC metrics which allow you to compare false positive and true positive rates. There are also conventional statistical techniques which can be utilized. The most significant metric for this study is the true positive detection rate. For a detector to be considered successful, a near 100% true-positive identification rate is desired. False positive detection rate is also important, but not as critical, to ensure the detector is not overly noisy.

3.2 Resources

We have access to a variety of computational resources from universities and research institutions across Europe. In Finland, the group has access to the CSC supercomputer. CSC provided supercomputer resources and hosted cloud services available for the project. Additionally, the researchers have access to limited computational resources via their personal computers.

3.3 Dataset Survey

In this section, 17 works focusing on streaming data outlier detection in machine learning are examined. The datasets used in each paper were examined to determine which datasets were commonly used in the literature as shown in Table 3.

Table 3. Datasets for Streaming Outlier Detection

Dataset	**Citation Count**	**Description**
KDD-CUP99[16]	6[17],[18],[19],[20],[21],[22]	Network intrusion detection dataset. Includes a wide variety of malicious and normal connections simulated in a military network environment.
Covertype-Forest[23]	4[24],[25],[26],[27]	Predicting forest cover type from cartographic variables only.
Shuttle[28]	3[29],[30],[31]	A multi-class classification dataset with dimensionality 9.
UCI-Vowel[32]	2[33],[34]	Nine participants spoke two Japaneese vowels sucesevily. 640 time series were created using linear predictions.
UCI-Pendigit[35]	2[36],[37]	Database of 250 hand-written digits from 44 participants.

[16]University of California, Irvine, 1999.
[17]Togbe et al., 2021.
[18]Na, D. Kim, and H. Yu, 2018.
[19]Salehi et al., 2017.
[20]Tan, Ting, and Liu, 2011.
[21]K. Yu, Shi, and Santoro, 2020.
[22]T. Kim and Park, 2020.
[23]UCI Machine Learning Repository, 1998a.
[24]Togbe et al., 2021.
[25]Salehi et al., 2017.
[26]Tan, Ting, and Liu, 2011.
[27]K. Yu, Shi, and Santoro, 2020.
[28]UCI Machine Learning Repository, n.d.(b).
[29]Togbe et al., 2021.
[30]Tan, Ting, and Liu, 2011.
[31]T. Kim and Park, 2020.
[32]UCI Machine Learning Repository, n.d.(a).
[33]Na, D. Kim, and H. Yu, 2018.
[34]Salehi et al., 2017.
[35]UCI Machine Learning Repository, 1998b.
[36]Na, D. Kim, and H. Yu, 2018.
[37]Salehi et al., 2017.

Figure 4 shows the most commonly occurring datasets in the literature from table 3 are KDD-CUP99 (University of California, Irvine, 1999) followed by Covertype-Forest (UCI Machine Learning Repository, 1998a). A dataset is included in Table 3 and Figure 4 if there are 2 or more occurrences in the literature surveyed.

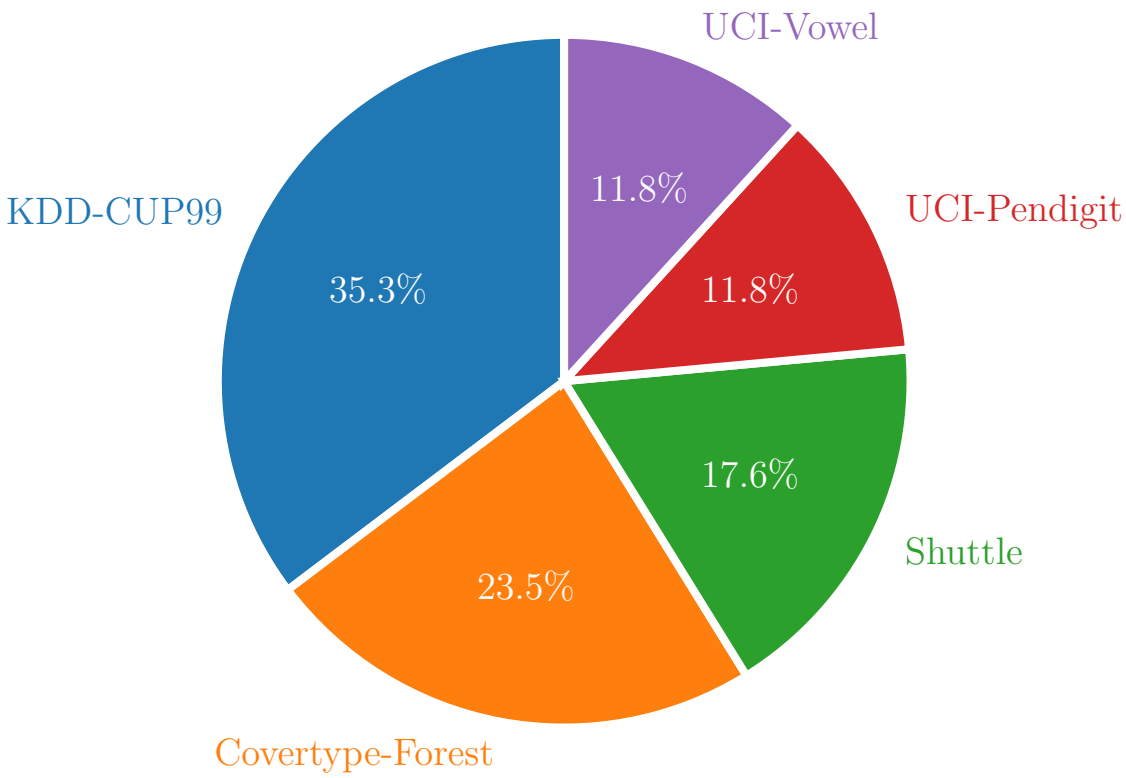


Figure 4. Dataset Occurrence in Literature

The most commonly used datasets found in the literature were originally created in the late 1990's. Most of the literature surveyed was conducted in the last 10 years, but the standard benchmark datasets are over two decades old. This data was collected when the landscape of cyber-attacks and computing looked much different than today. These datasets are not sufficient to benchmark performance against modern, sophisticated attacks. There is a lack of standardization in datasets that makes compareing algorithm performance difficult.

By analyzing the existing datasets, the experimental datasets for this study are selected. The current industry standard datasets are not sufficient for benchmarking performance in relation to the use cases presented in this study. In the subsequent section, appropriate and diverse datasets are selected that benchmark the desired outlier phenomenon.

3.4 Dataset Selection

In this study, three separate datasets are used to test and evaluate the developed detection technique. The datasets span many disciplines and provide a good survey of the applicability of the proposed detection technique.

3.4.1 Hydraulic Simulation Dataset

For this experiment, the Matlab software toolkit Simscape Multibody is used to design a model of the hydraulic system outlined in Figure 5. This model is used to study the response of the system to optimize its behavior and parameters. For this study, a mass (m) of 240 kg and a supply pressure of 185 bar is used.

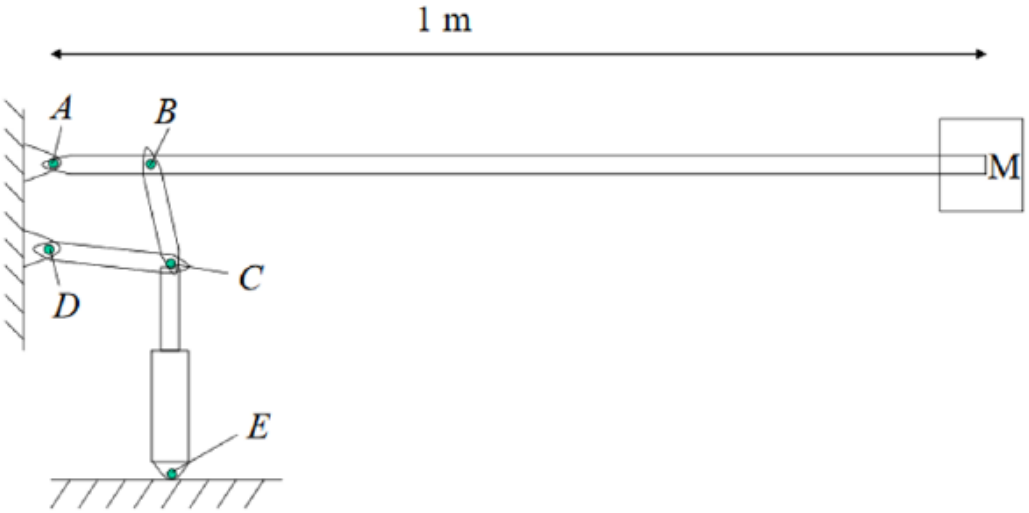


Figure 5. Hydraulic Boom Lift Structure

The square wave profile control signal shown in Figure 6 is used as input to the simulation of the hydraulic system. It stays at a neutral voltage of 0 until 0.2 seconds into the simulation. At 0.2 seconds, the control signal is at its maximum of 10 volts for 0.4 seconds. Then it falls to become proportionally negative at 0.7 seconds. After another 0.4 seconds elapsed, the signal returns to the neural position of 0 volts beginning at 1.1 seconds.

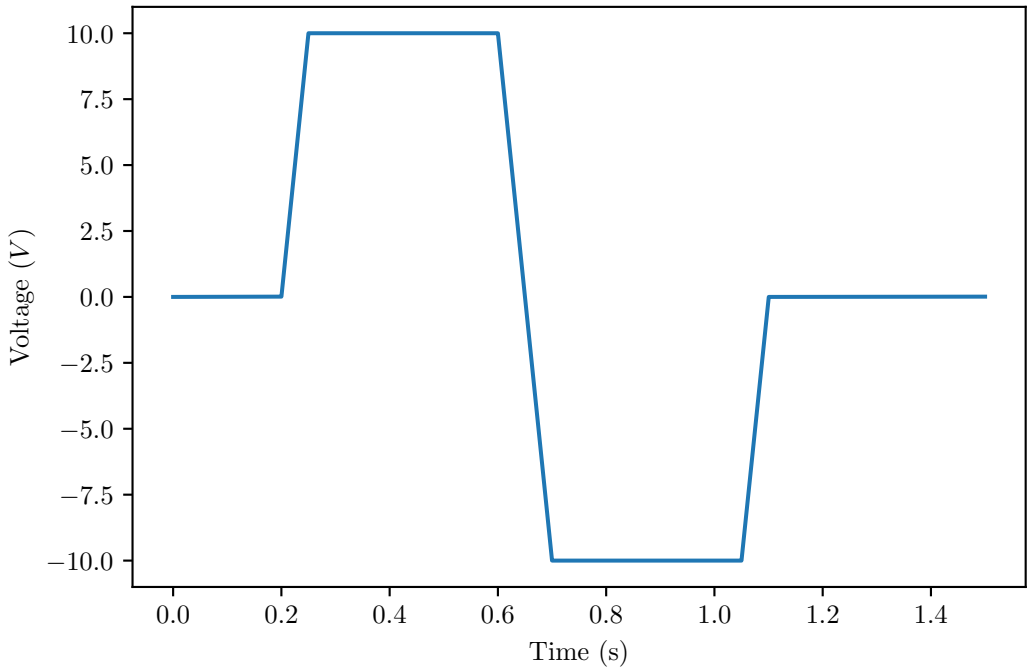


Figure 6. Hydraulic System Control Signal

Figure 7 shows that introducing a proportionally negative signal does not cause the end effector to return to its initial position. As the control signal is varied inversely, the position of the end effector only returns halfway to its initial position. This indicates the system exhibits a non-linear relationship between the control signal and the end effector position. The system experiences mild oscillation after coming to rest when the control signal returns to zero at the end of the simulation. To achieve predictable system response and reduce oscillation, a more complex control methodology is required.

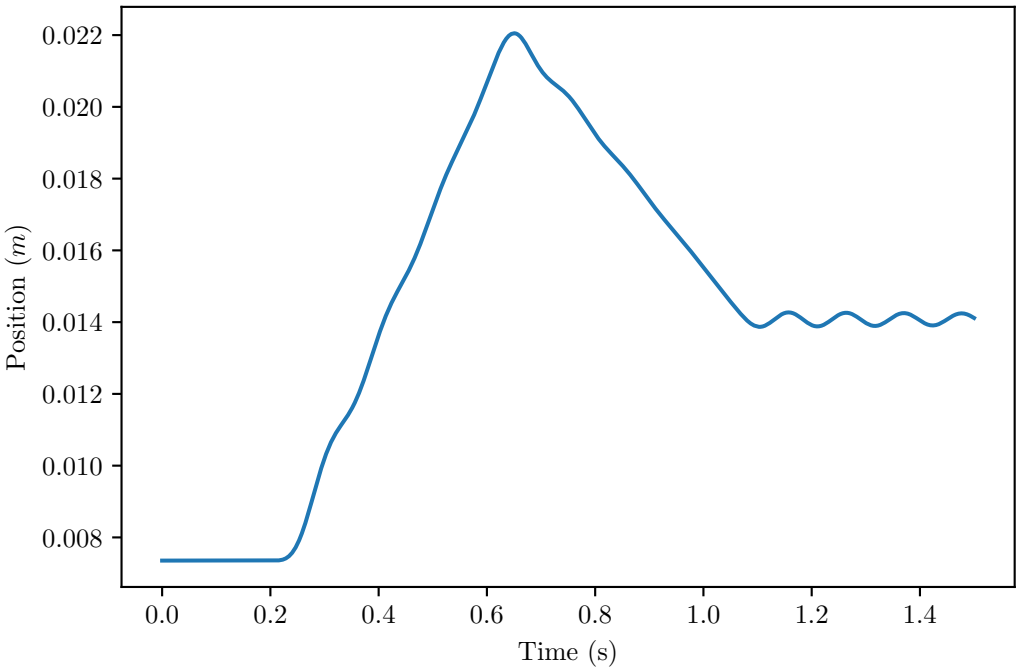


Figure 7. Hydraulic Crane End Effector Position

3.4.2 Power Electronic Converter Dataset

Transitioning from conventional power systems to power electronics-dominated grids (PEDG) has increased demand for grid-forming converters (GFM) to facilitate operational reliability. GFMs have made significant progress in recent years to expedite stability under different grid conditions, but their operation during faults or large signal disturbances still remains a challenge.

Authors Rokrok et al., 2022 explain that GFMs handle a significantly smaller percentage of over-current (usually only 20%) compared to synchronous generators (SGs) which can handle seven times their nominal current. This makes fault detection for GFMs critical to maintain synchronization with the grid. Since the network infrastructure of power systems keeps expanding, it is important to identify these faults accurately under varying grid parameter uncertainties.

This study examines four faults in the PEC dataset. The frequency $[f_c]$ of the system throughout various fault conditions is used for detection. Each fault has significantly different characteristics and magnitude. The faults examined in the study are explained in Table 4.

Table 4. PEC Dataset Classifications

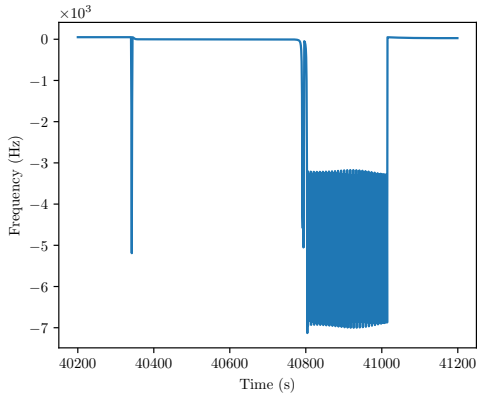
Fault	**Description**
Line-to-Line (LL) Fault	An LL fault is also referred to as an unsymmetrical fault and occurs when there is a short circuit between two conductors. In three phase power, this can occur between two phases of the system. This fault causes a significant decrease in frequency that is orders of magnitude greater than the standard frequency of the system.
Three-Phase Sensor Fault	During a three phase sensor fault, there is nothing wrong with the system itself but there is faulty sensor in the system. This causes the detected frequency to rise slightly. This slight rise is significantly less than the other examined faults and is very close to the reference frequency of 50 Hz.
Single-Phase Voltage Sag	During a single phase voltage sag, the frequency oscillates continuously until the fault is over. This is a significant fault in the system and detection is critical to take remedial action. This is another fault where the magnitude is not very large in comparison to the LL Fault of the Three Phase Grid Fault.

Continued on next page

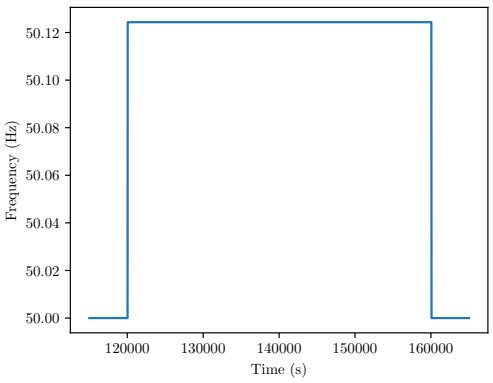
Table 4 – *Continued from previous page*

Fault	**Description**
Three-Phase Grid Fault	A three phase grid fault is a severe fault where there is a problem with the grid and corrective action needs to be taken promptly. In this fault there is a large magnitude drop in frequency for the duration of the fault. This is similar behavior to the Three Phase Sensor Fault but the frequency change of the difference is orders of magnitude larger and in the negative direction.

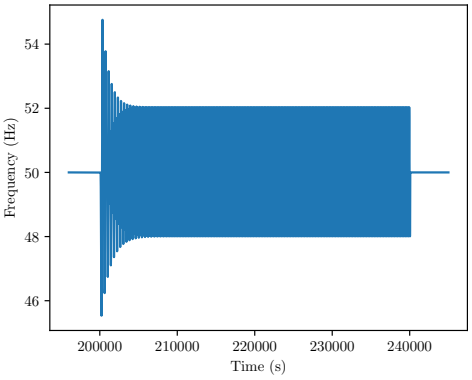
The faults explained in Table 4 are presented visually in Figure 8.



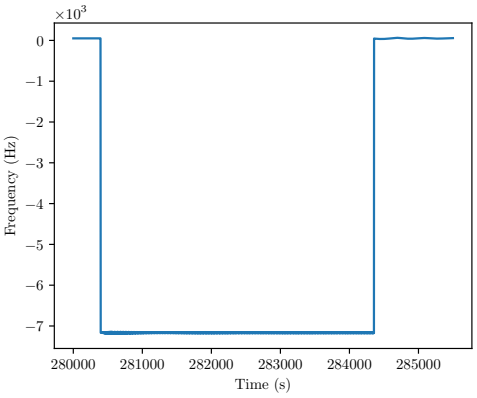
(a) Line-to-Line (LL) Fault



(b) Three-Phase Sensor Fault



(c) Single-Phase Voltage Sag



(d) Three-Phase Grid Fault

Figure 8. PEC Dataset Fault Visualization

3.4.3 Cyber Security BETH Dataset

On the Github Discussion Board (Beattie, 2021) for the popular River (Montiel et al., 2020) Machine Learning library, I proposed using a streaming Local Outlier Factor (LOF) methodology for outlier detection. Through this discussion, Authors Highnam et al., 2021, added the BETH dataset to a public repository at Imperial College London which provides easy access for researchers wishing to work on the dataset. The LOF method is unsuitable for contextual outlier detection and is not selected for implementation in the River library.

The BPF-extended tracking honeypot (BETH) cyber security dataset (Highnam et al., 2021) was released in 2021. It is still under active development and testing. A cybersecurity honeypot is a set of computer resources that an organization actively monitors for intrusion detection. Hackers benefit from exploiting these honeypots but are usually unable to do harm to any production systems. This makes it an attractive dataset for this study since it reflects the current state of the art in cybersecurity and is one of the first to be designed for uncertainty analysis and anomaly detection.

For this dataset, an ssh vulnerability is exploited where any password entered allows a user to login. The system is running two auxiliary containers to monitor traffic. The first is the Berkely Packet Filter (BPF) which examines OS process management calls. The second monitor logs DNS activity from the system. This data is collected and parsed over a series of trials to form the dataset.

Authors Highnam et al., 2021 present a number of advantages of this dataset that make it attractive for modern machine learning and anomaly detection research. This includes: (i) being a large and comprehensive cyber-security dataset (ii) containing modern attack vectors (iii) including benign and attack information for each host that is fully labeled.

Figure 9 shows the samples from the BETH dataset for the UserID parameter that is used in this study. This parameter is treated as a continuous data stream that corresponds with the measurement time. The spikes in the UserID parameter generally correspond with the labeled outliers.

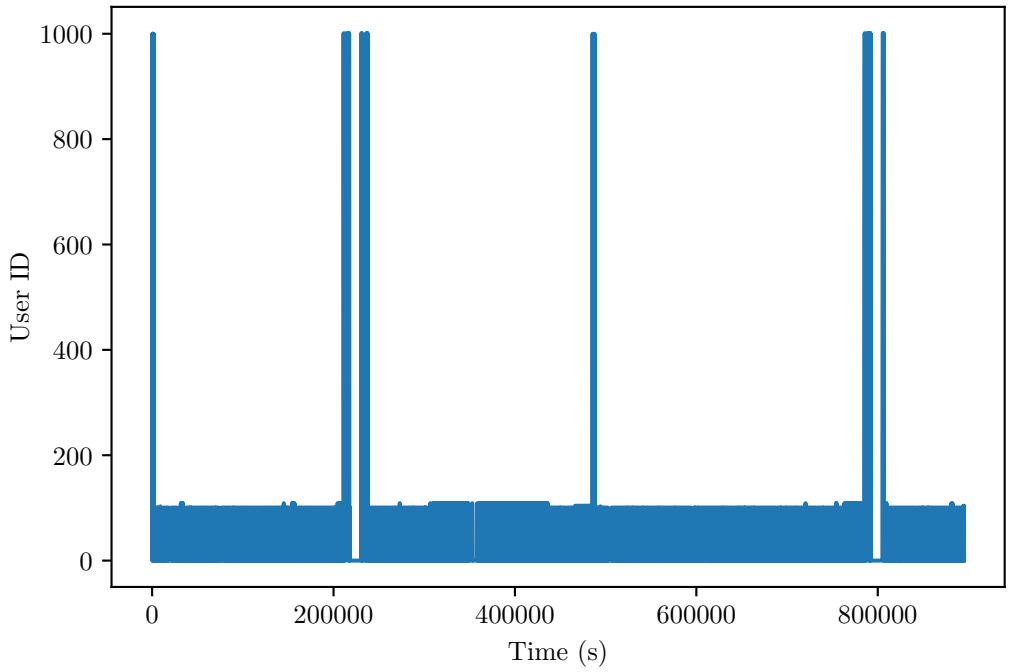


Figure 9. BETH Dataset Signal

In the BETH dataset, the honypots are deployed in a cloud environment because many major companies utilize cloud computing resources. Therefore it is important to understand the attacks that are specifically scanning the cloud provider space compared to the prior datasets which are collected exclusively through on-premise servers. Typically a dataset is selected to: (i) benchmark and highlight a technique in a specific area or (ii) demonstrate the capabilities and generalizability of a technique to other disciplines. The BETH dataset is designed to be useful and easy to work with for machine learning researchers. It provides large volumes of fully labeled data with an easy to parse format and understandable data points. In this work, the anomaly detection problem is approached differently than in traditional machine learning approaches which shows the generalizability of the dataset.

3.5 Algorithm Development

The code development and experimentation described in this work is available on a public GitHub repository (Beattie, 2022). The figures and results presented in Section 4 are generated using this code. This section outlines how the final outlier detector was developed and some of the unsuccessful detection attempts. Some of the datasets in the repository are private and as such cannot be included in the repository. The code to perform the analysis is included in the repository and can be adapted to fit similar datasets or problems.

3.5.1 Unsuccessful Attempts

Many libraries and algorithms outlined in Section 2.7 are tested in this study. The first attempts utilized techniques from the river ML (Montiel et al., 2020) and pysad (Yilmaz and Kozat, 2020) libraries. The tested algorithms in the pysad library include IForestASD, LODA, RSHash, xStream, and Robust Random Cut Forest. The tested (and only) algorithm from the river library is Half Space Trees.

The algorithms demonstrated good performance for point-wise outliers on a simulated noisy sine wave. The algorithms performed poorly against the contextual outliers in the Power Electronics Dataset. The outliers present in the dataets in this study are not pointwise but contextual shaplet outliers. This shows that the detection techniques for point-wise outliers are not sufficent for detecting contextual outliers. A review of the outlier taxonomies discussed here is presented in Section 2.4.

New techniques are selected that meet the following criteria: (i) do not require training of a model or previous knowledge of the data (ii) operate in a streaming context (iii) use a windowed or fix memory and processing allocation. This significantly reduces the list of available algorithms and libraries. Many require model training and do not operate well in non-batch contexts. The banpei (Tsuruta and Feng, 2021)

library's Singular Spectrum Transformation (SST) and Hotelling methods also failed to detect the desired anomalies. The Gradient and Difference detectors in TSOD (Andersson, Mariegaard, and Falk, 2021) are also unable to detect the contextual anomalies.

3.5.2 Matrix Profile Detector

The Matrix Profile algorithm for contextual outlier detection from the STUMPY library (Law, 2019) was selected for implementation. Initial testing with the multivariate matrix profile demonstrated good performance but was very computationally expensive and slow. Additionally, the library did not facilitate an iterative computation for this method so each iteration would need to recompute the entire matrix profile across the whole window.

Because of this, the univariate matrix profile is used in this study. It is faster and allows for iterative updates to the matrix which is significantly more efficient. An outlier model class is created to compute the matrix profile for each iteration and predict whether the current event is an outlier.

This technique is used in Section 4 to detect outliers in a variety of situations. It requires adjusting three parameters based on the characteristics of the data being analyzed. This is less than the number of parameters required by most machine learning techniques. Additionally, it relies strictly on statics computations and is therefore explainable without reverse engineering.

$$\epsilon_{[i]} = \max_{mp} \geq (\mu_{mp} + (\sigma_{mp} \cdot \alpha)) \tag{1}$$

Let α be the standard deviation (σ) multiplication factor in Equation (1). This equation is used to calculate anomalies by determining the current mean value and comparing it against a multiplier of the standard deviation of the matrix profile. This equation is parameterized in code so the user can set the data window size, analysis window size, standard deviation multiplier, and other parameters to tune detection. Optional filters are included to avoid triggering while the detector is still loading data into the analysis window and on recent faults to avoid redundant detection.

3.5.3 Detection Filters

$$\lambda_{[i]} = \left| \epsilon_{[i]} - \max_{mp} \right| > 0.01 \tag{2}$$

Equation (2) shows a filter for the detector to ensure that the currently calculated metric (ϵ) is representative enough to trigger a detection. If this condition is not true, there is not enough information to conclude definitively that the point is an outlier. Therefore, it should not trigger detection.

In certain datasets, like the BETH dataset, performance was inadequate without filtering. A significant number of false positives were detected before the implementation of the rolling range filter.

$$\gamma_{[i]} = \max_{mp} - \min_{mp} > \left(\mu_{\epsilon[(n-5)\ldots n]} \cdot \beta \right) \tag{3}$$

In Equation (3), the rolling range average (μ) is multiplied by a scaling factor (β). The range in the window is computed during every iteration by subtracting the maximum and minimum value. A ring buffer is implemented to store the rolling range values. In this buffer, only 5 elements are allowed and when a new element is inserted the last element is removed. This means that only the 5 most recent elements are available at a time.

To be stored in the buffer, the current range must be greater than the average of the rolling range ring buffer multiplied by the standard deviation multiplier. If this is the case, the current range is appended to the rolling range ring buffer and is considered an anomaly.

3.5.4 Triggering Detection

Using the matrix profile and the filters developed above, each data point is evaluated and determined to be anomalous or normal.

$$\text{anomaly}_{[i]} = \begin{cases} true, & \text{if } \epsilon_{[i]} \wedge \lambda_{[i]} \wedge \gamma_{[i]} \\ false, & \text{otherwise} \end{cases} \tag{4}$$

The anomaly level is calculated using Equations (1), (2), and (3) respectively. To be determined anomalous, the point (i) must satisfy all the conditions in Equation (4). If a point does not satisfy these conditions, it is not considered anomalous and it subsequently does not trigger the detector. The combination of the matrix profile algorithm and targeted filters enable the developed detector to be flexible and robust to a wide range of datasets and window sizes.

4 Results

The results from the literature reviews and the development of the anomaly detector are presented in this section. The anomaly detector is created in Python with the STUMPY library implementation of the iterative matrix profile. The datasets from Section 3.4 are tested here using the developed detector against different properties and anomaly types. The parameters for each trail are listed and the results are explained. Further analysis is performed in Section 5.

4.1 Hydraulic Simulation Dataset

In this section, the hydraulic simulation dataset presented in Section 3.4.1 is tested with the parameters in Table 5 using the developed anomaly detector. The hydraulic dataset is the smallest dataset in this study and it outlines the performance and characteristics of the anomaly detector when using small window sizes.

In this experiment, the standard deviation multiplier is set to two for the detector. If the data follows a standard Gaussian distribution, the detected points would fall outside 95% of the data present in the window. This would represent a high chance of a motif change at this small window size. The rolling range detector is set similarly to a value of two to weight recall equally with present results. The recent range detection debounce multiplier is also set to the standard value of two to ensure that there are not overly-noisy or duplicate detections.

This experiment demonstrates that the developed detector is robust to changes in scale. Figure 11 shows that the pressure and force signals in the system have similar shape but different magnitude.

$$F = (A_1 \cdot p_A) - \dot{x}|A_1 \cdot p_A|(1 - \eta) \tag{5}$$

This relationship in the shape of the pressure and force signals is derived from Equation 5. As the pressure increases, the force exerted by the hydraulic cylinder increase proportionally. Since the boom arm and end effector in the system are fixed, the end effector moves proportionally to the change in force. Figure 11 shows small differences between the force and pressure which is a result of the modeling of the hydraulic boom arm as a flexible body, which introduces oscillation that affects system pressure.

Table 5. Hydraulic Simulation Detector Parameters

Parameter	**Value**	**Description**
m	15	Window Size
ts_size	30	Time Series Size
std_dev	2	Standard Deviation Multiplier
range	2	Rolling Range Multiplier
recent	2	Recent Detection Debounce

Figure 11 shows the detector was able to locate all of the motif changes present in the signal which represent inflection points. This is a 100% accuracy rate for this experiment without any false positives. The amplitude and waveform of the signals presented here are identical with the primary difference being in the signal amplitudes. The pressure signal is two orders of magnitude larger in amplitude but still triggers at the same time as the force signal. This illustrates the detector is robust to scale variations.

The shape of the matrix profile values in Figure 10 are nearly identical for the force and pressure values. The two signals have dramatically different magnitudes but similar shapes. This shows the algorithm is robust to different macro data scales and explains the results obtained in Figure 11.

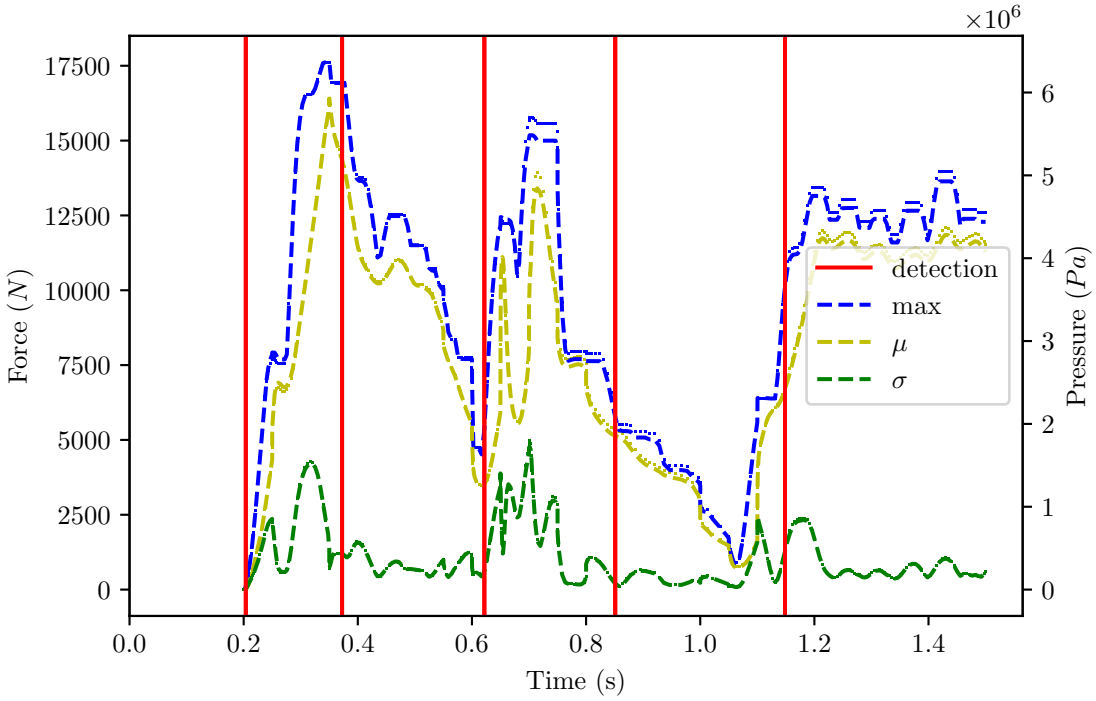


Figure 10. Hydraulic Simulation Matrix Profile Values

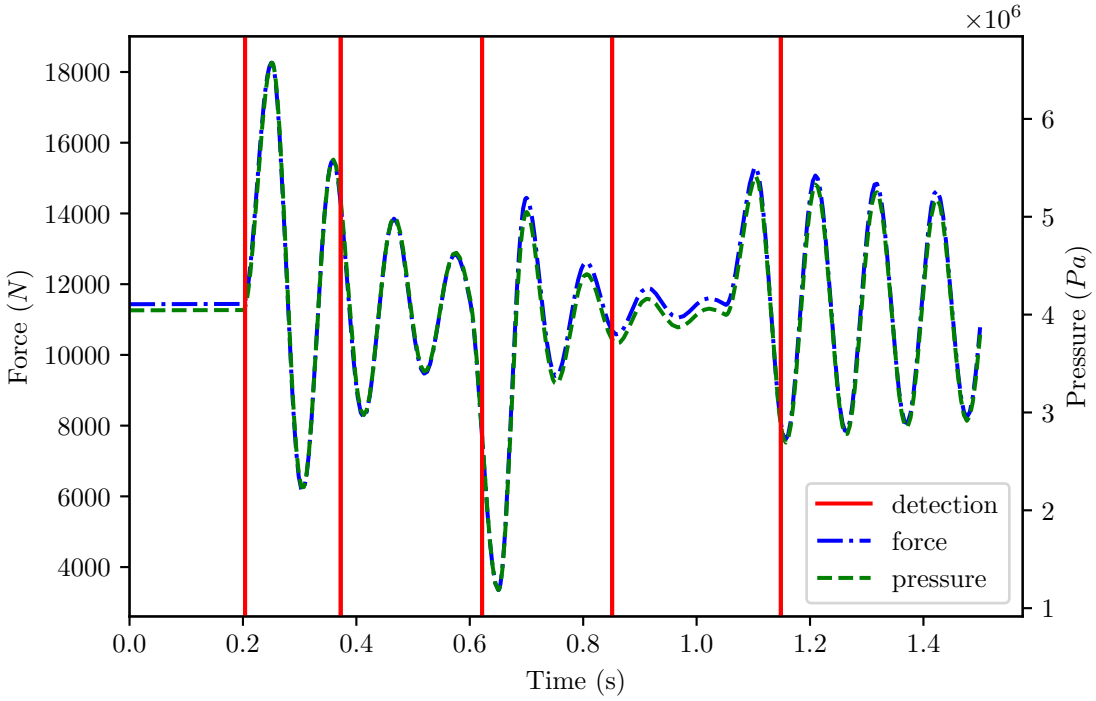


Figure 11. Hydraulic Crane Motif Changes

4.2 Power Electronic Converter Dataset

In this section, the Power Electronic Converter (PEC) dataset is tested with the parameters from Table 6 using the developed anomaly detector. The PEC dataset is significantly larger than the Hydraulic Simulation dataset, containing approximately 300,000 time steps (83 hour of signal data). The PEC dataset outlines the scenarios at which this detector excels, and highlights its performance with medium window sizes.

The standard deviation multiplier in this trial is significant. If the data follows a standard Gaussian distribution, the detected points would fall outside 99.999% of the data present in the window. This would represent a significant outlier comparative to the rest of the data.

The rolling range multiplier is disabled for this experiment since the outliers are not relational to each other. There are 4 different types of outliers which all have radically different behaviors and signal shapes, so it is not desired to compare or remember them in the context of the next outlier. The recent range detection debounce multiplier is set to one due to the larger size of the time series window and overall window size.

Table 6. PEC Dataset Detector Parameters

Parameter	**Value**	**Description**
m	250	Window Size
ts_size	5000	Time Series Size
std_dev	4	Standard Deviation Multiplier
range	0	Rolling Range Multiplier
recent	1	Recent Detection Debounce

Because of the scale variance of the anomalies, only 2 of the 4 detected anomalies are visually represented in Figure 12. The two anomalies that are not depicted show similar patterns, but at much smaller scale. This shows the detector is able to perform on signals with widely differing anomaly characteristics and patterns.

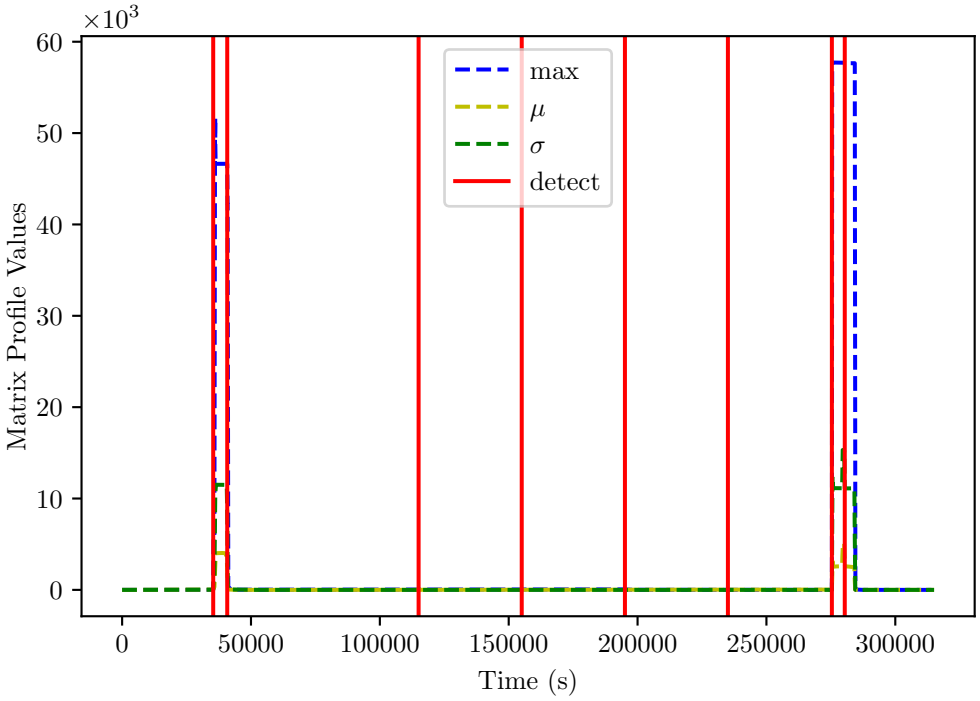
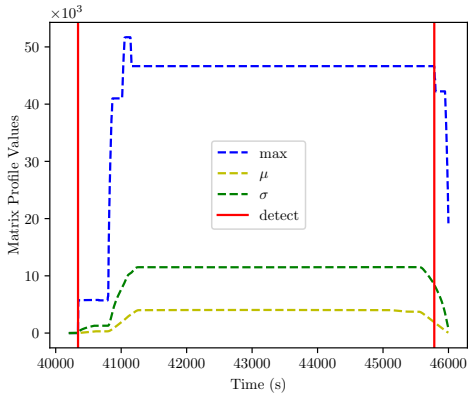
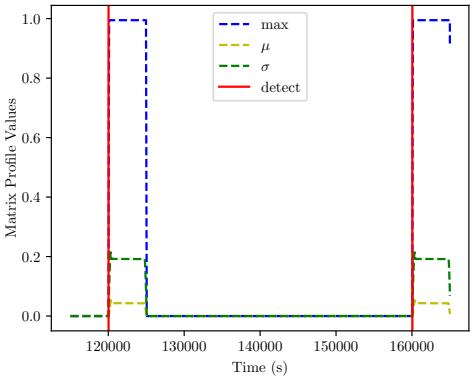


Figure 12. PEC Dataset Matrix Profile Values

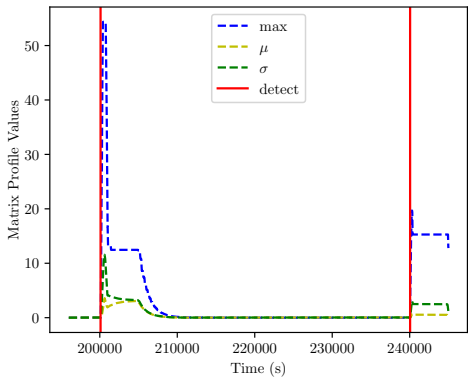
Figure 13 shows a zoomed in window of the matrix profile values during each fault type examined in this experiment. A description of the characteristics of these fault types is found in Section 3.4.2. The detector behaves robustly and accurately for all the fault types presented.



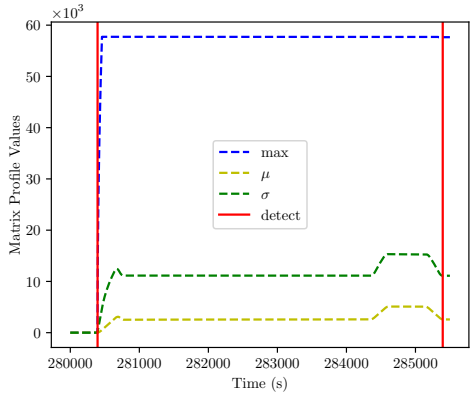
(a) Line-to-Line (LL) Fault



(b) Three-Phase Sensor Fault



(c) Single-Phase Voltage Sag



(d) Three-Phase Grid Fault

Figure 13. PEC Dataset Fault Matrix Profile

Figure 14 shows the detector accurately determined the start and end of each anomaly (100% detection rate) with no false positives (0% error rate). It performs this detection within 1 window size (from Table 6). Although detecting the ending of an anomaly sometimes takes more time than detecting the start.

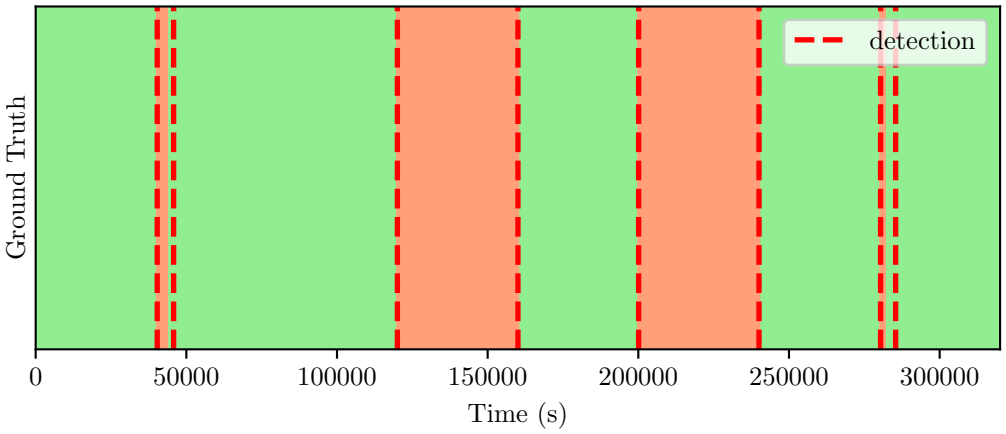


Figure 14. PEC Ground Truth Comparison [Normal: Green, Anomaly: Red]

By nature, anomalies are rare so interpreting the detection and false positive rates in a traditional way is not advisable. The interpretation of the results of this trial are presented in Section 5.3.

4.3 Cyber Security BETH Dataset

In this section, the BETH dataset from Section 3.4.3 is tested using the parameters in Table 7 with the developed detector. The BETH dataset is the largest in the study, containing approximately 800,000 time steps (9 days of signal data). It outlines the most challenging test scenario presented for the detector in this study and highlights its performance on large window size data.

If the data follows a standard Gaussian distribution, the detected points would fall outside 99.7% of the data present in the window. The rolling range multiplier is set to one below the standard deviation multiplier for this experiment. This enables recall of previous outliers, but more heavily weights the present outlier detection score. The anomalies in this dataset have been manually labeled and although there are different kinds of attacks, the baseline signal (userId) remains the same. The recent range detection debounce multiplier is set to two due to the nature and volatility of the data.

Table 7. BETH Dataset Model Parameters

Parameter	**Value**	**Description**
m	1000	Window Size
ts_size	10,000	Time Series Size
std_dev	3	Standard Deviation Multiplier
range	2	Rolling Range Multiplier
recent	2	Recent Detection Debounce

In this study, the outliers are only present when there is a significantly large range spike in the matrix profile. An example of this is shown near time step 200,000 in Figure 15. The regular and periodic small peaks in the data are not outliers and the rolling range technique ensures they do not trigger the detector. This illustrates how the rolling range filter can be used to tune the memory and accuracy of the detector.

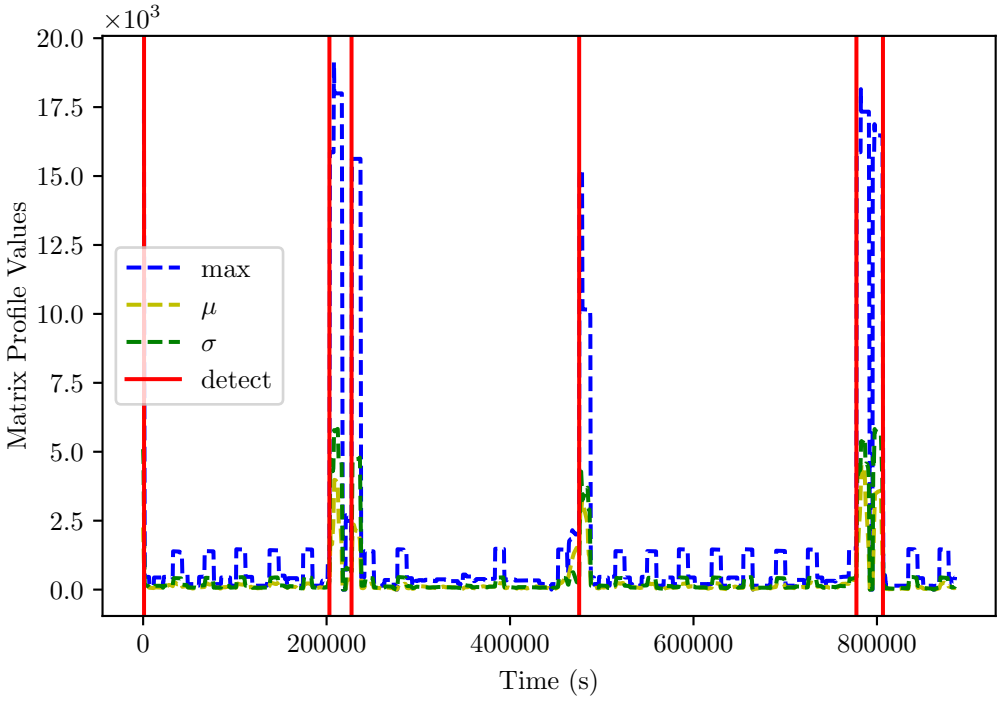


Figure 15. BETH Matrix Profile Values

Figure 16 illustrates that the algorithm was able to detect the starts of the outliers within the debounce threshold and detect the faults' endings with some delay. There are 2 false positives from the detector, which correspond to high UserID values in the dataset. These two locations are marked as suspicious so in this case the false positive is worth investigating even if it is not labeled as dangerous.

Authors Highnam et al., 2021 determined that many of the data points present are considered suspicious. Figure 16 shows that only a few of the suspicious outliers are actually dangerous.

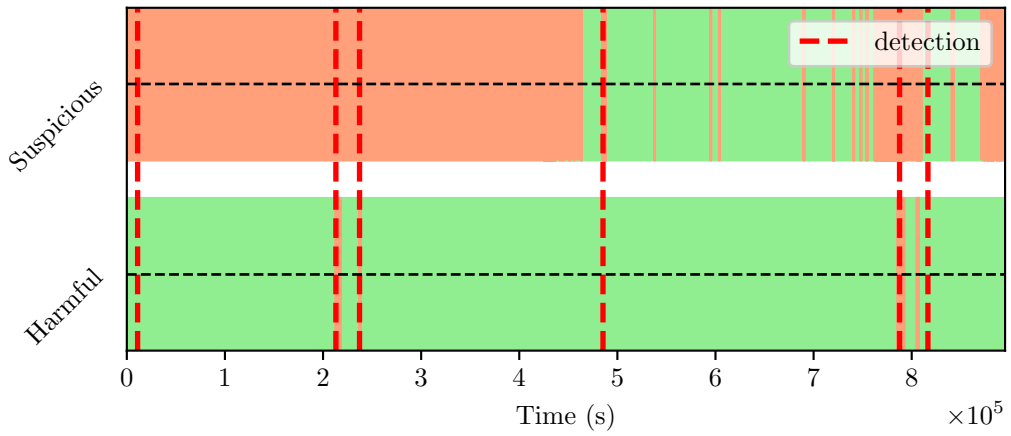


Figure 16. BETH Ground Truth Comparison [Normal: Green, Anomaly: Red]

In this experiment, the detector is able to discern where the dangerous outliers lie among the very noisy suspicious data points. The final fault that appears as two spikes is part of the same attack and can be classified as one fault instead of two. While this is the most challenging dataset analyzed, the detector still provides a 100% true positive identification rate. For this experiment, the false positive detection was higher than in the other experiments. This is problematic, but not critical, as a false positive can be manually examined and ultimately ignored.

5 Discussion

This section provides analysis and discussion of the results presented in Section 4. Following the analysis of each experiment, further developments and future applications are proposed to continue this work and implement it in other contexts.

Outliers and anomalies by nature are rare and fall significantly outside the normal distributions of data. Traditionally it is desired to minimize or eliminate outliers and anomalies in data to provide more regular and predictable performance. When focusing on outliers and anomalies given these characteristics, conventional algorithm metrics are not particularly helpful to analyze the success of a detection algorithm.

In the cases in this study, a false positive does not carry the same implication as in a machine learning algorithm. In this case, a detection triggers further examination and an undetected positive carries a much higher consequence. If a fault occurs and is not detected, the operator of the system would be unaware and there could be significant consequences or damage to the system.

For this reason it is more important to evaluate the true positive detection rate, which in all the experiments is 100%. Additionally, the developed technique did not provide many false positives (only occurring in the BETH dataset) and was not noisy which provides a high degree of reliability. Utilizing this reference frame, the developed algorithm was very successful in identifying outliers in a wide variety of contexts.

5.1 Problems with Existing Datasets

In this section, analysis of the datasets surveyed in Section 3.3 are presented. Additionally, discussions with industry experts on the shortcomings of the 'standard' datasets for modern applications are summarized and presented.

Most papers surveyed included these seemingly standard datasets while also utilizing a wide variety of custom, private, or less well known datasets. The existing standard datasets are insufficient to benchmark performance for modern streaming outlier detection algorithms.

Some practical issues with the most popular cited dataset, KDD-CUP99 include: (i) the landscape of cyber-attacks has significantly changed since 1999 (when KDD-CUP99 was published); (ii) Computer and network architectures are different with new technologies like Web3; (iii) Modern network protocols are not included in older datasets; and (iv) Logging of network requests and data has changed significantly.

Modern datasets also face shortcomings. There are some practical issues with the BETH dataset including: (i) The data was collected across many days, but only for a short period of consecutive time, around 5 hours. (ii) The SSH vulnerability exploited in the dataset allows unsophisticated attackers access if they entered any password. (iii) Only DNS, not comprehensive system logs, are available. A standard test suite or dataset should be created to uniformly evaluate performance of different algorithms and techniques.

5.2 Hydraulic Simulation Dataset

In this dataset, there are not faults specifically but instead macroscopic changes in the signal. The algorithm effectively detected all these macroscopic changes for both the original and modified control signal. This is a very small dataset comparatively and the detector is operating on very small window sizes and consequently very little data in each window. This indicates that the technique is effective at various window sizes from small to large. The implications of the detector in mechatronic systems is wide ranging and is described in the next section.

The system has oscillation after it returns to the resting position as shown in Figures 7 and 11. The hydraulic fluid used in the system is acting as a spring which is what causes the oscillation.

$$k = B_e \left(\frac{A^2}{V} \right) \tag{6}$$

Equation (6) models the spring constant for the system where V is the volume of the hose and cylinder, A is the surface area of the piston and B_e is the effective bulk modulus.

One option for reducing oscillation is increasing the spring constant $[k]$. Analyzing Equation (6) shows this could be done by increasing the piston area, decreasing the hose and cylinder volume or increasing the effective bulk modulus. There are physical limits of the hydraulic fluid selected and other practical considerations that make this challenging. Selecting another hydraulic fluid can be complex, time consuming, and expensive so this is very uncommon after the system has been implemented.

Another option to reduce oscillation is to introduce dampers to the system. This component would need to be tuned correctly and then periodic oscillation could be reduced. A dynamic damper could also be implemented that used the output of the algorithm to dynamically adjust the damping coefficient for different vibration cases

Adding a control unit that combats oscillation by dynamically adjusting system pressure is another option for more robustly reducing oscillation. A Proportional Integral Derivative (PID) control loop could be introduced to tune the system response and eliminate response thrashing. The PID controller can also be tuned to

achieve a desired system response and offer more consistent and reliable control. This PID controller can be coupled with the algorithm result to determine the start and end of oscillation in the signal and to progressively adjust the PID values to create a dynamic PID controller.

5.3 Power Electronic Converter Dataset

The detector was able to successfully detect all four faults with zero false positives in the PEC dataset. In the case of the power grid, it is important to detect anomalies quickly so that remedial action can be taken quickly. The detection of the event happened quickly and was usually within half of one window size or less than 125 time steps.

The detector preformed well when faced with anomalies that have dramatically different characteristics and magnitude. Each anomaly in the dataset is shaped significantly differently and the algorithm accurately detected the start and end of each one. This illustrates the robustness to scale of the algorithm.

Additional anomalies which are not considered faults are present in the PEC dataset. In further experimentation, these anomalies can be examined and the algorithm can be tuned to detect them as well. Further experimentation could also be conducted using the multi-variate matrix profile on the three phase current or voltage inputs to detect anomalies. This was not preformed in this experiment because it is computationally much more expensive and the library does not currently offer a way to compute the multi-variate matrix profile iteratively as with the uni-variate matrix profile.

This algorithm was also designed to be run in real time and was simulated as such. Each datapoint arrived in a simulated time series so that the algorithm was not aware of subsequent datapoints. It also utilizes a sliding window technique so it fits in a fixed memory window. Additionally the computations are efficient and scale well with the window size. This gives the algorithm strong potential in a real-time monitoring contexts.

In the future, the detection system can be implemented in MATLAB and used in a real-time monitoring and detection application. There is currently a MATLAB power grid control and monitoring interface for a real system and in order to integrate with it, a module for MATLAB must be created. This would involve creating an implementation of the matrix profile algorithm in MATLAB and utilizing that algorithm in tandem with the control system.

With this setup it would be possible to test the real-time detection capacity of the algorithm. Once this is implemented, it would be possible to couple it with a classification algorithm to attempt to determine the type of fault. If a fault was determined, the appropriate correct action could be preformed depending on the classification.

Real-time detection and classification has significant implications in grid cyber-security and reliability. Attacks on the power grid are becoming more sophisticated and it is important to know whether the system is under attack or experiencing a fault condition. Furthermore, it is important to know that a fault is occurring so that automated or manual corrective action can be preformed so there is no damage to important system components and to ensure maximum system reliability and up time.

5.4 Cyber Security BETH Dataset

As the most complex and largest dataset, the BETH dataset presents challenges to any anomaly detection technique. The matrix profile anomaly detector presented here performed well, not missing any of the evil outliers. When the detector signals an anomaly, that can trigger a set of automated actions as well as a manual audit of the system to examine malicious or suspicious behavior around the time of detection.

This technique can also be used to examine past or historical data to identify outliers. The matrix profile technique can be fine tuned even further when the entire window or dataset is known at once, which can benefit researchers analyzing previous attacks.

The largest improvement for this dataset would be to include multiple data streams in the detector. Currently the detector is only analyzing one of the over 10 data streams available in the dataset. By adding the ability to process additional streams, the algorithm could be tuned for higher accuracy and performance. Adding more streams is challenging because the multi-variate matrix profile is computationally expensive and currently not iterative. A solution to this could be to run multiple uni-variate matrix profiles in parallel and aggregate the results into a unified metric.

In cyber-security, it is critical to detect and stop attacks as soon as possible. This detector can be developed to provide a real-time intrusion and monitoring system. Various system metrics can be collected and fed to the detection pipeline. If an anomaly is detected, the system outputs a warning and appropriate handlers can be configured to monitor and mediate the intrusion. This system can help generate less false positives than current monitoring solutions and provide real-time insights into potential system intrusions.

5.5 Future Algorithm Development

There are many possibilities to develop the algorithm further, and combine it with other detection techniques. One possibility is to convolve the detector for different window sizes. This would entail multiple detectors running in parallel with different window sizes to detect macro and microscopic changes in the signal. This would provide valuable insight about the system and illustrate how microscopic changes do or do not influence the overall system behavior.

From an algorithm standpoint, there are many opportunities to improve performance and usability. Implementing the detector for use in other languages allows for broader future implementations, like in the PEC use case. For use in real-time monitoring, detection methodologies have to be closely coupled with the system they are interacting with. For this reason, implementation of theoretical algorithms and concepts is key to the successful adoption of the techniques presented here and in other works.

This detector can also be combined with other machine learning techniques to form a pipeline for more specific detection applications. For example with the PEC dataset, this detector can be combined with a categorical detector that is trained to detect various common faults usually present in the power grid. The anomaly detector signals the beginning and end of a fault, and during a fault, the categorical detector classifies the fault.

Then the appropriate action can be taken to rectify the fault to ensure reliability and up-time of the grid. If the categorical detector is unable to classify the fault, it could be a cyber-attack or rare fault that requires manual intervention and overview. In this case, the relevant system operators can be notified and take corrective action to avoid possible catastrophic system failure or compromise because of an undetected intrusion or problem.

5.6 Open Questions

The results presented in this work lay the foundation for further experimentation in time-series anomaly detection in production systems. Further open research challenges include:

- How can we efficiently and robustly compare the results of diverse algorithms and techniques (*e.g.,* streaming, batch, statistical, streaming, etc.) against test datasets?
- How can different anomaly detection techniques (*e.g.,* statistics, deep learning, etc.) be combined (ensembled) to increase overall prediction capability and accuracy?
- How can the developed detector be improved to include multi-variate prediction and anomaly notification?
- How can the developed detector trigger preventative measures to mitigate the impact of anomalous or adversarial data?
- How can the developed detector be incorporated into a production system for a monitoring and testing pilot to improve real-time monitoring and anomaly detection?

Combining the algorithm improvements proposed in Section 5.5 with the open research questions presented above, further development will create solutions that demonstrate the practical value of this work and the FIREMAN project. Subsequent work will focus on using the detection methodologies presented in this work and implementing them in pilot real-time monitoring applications through the FIREMAN project.

6 Conclusion

This work begins by providing an introduction to the research objectives and overall structure of the FIREMAN project. This work is principally focused on identifying detection strategies for real-time anomalies in streaming time series data. Two broad outlier taxonomies are presented with three sub-taxonomies for the contextual outliers present in the experimental datasets. These outliers focus the research problem to context-wise shaplet outlier detection.

A review of modern anomaly detection algorithms and libraries in Python is presented in Section 2.6 and 2.7 respectively. This review evaluates the strengths and weaknesses of the libraries as well as their applicability for use in building an anomaly detector. This review provides the foundation for the development of the anomaly detector utilizing the Stumpy Matrix Profile library and detection filters in Section 3.5.2.

A survey of the most common datasets used in the literature is presented in Section 3.3. This survey found the KDD-CUP99 dataset is still very commonly used, even though it has many shortcomings and is over 20 years old. This is problematic since cyber-attacks have changed dramatically since then and now have a different ontology and language.

The author contacted industry experts to understand how current datasets are used and why. These discussions led to the selection of the BETH dataset for one of the experiments. It presents an extensive set of labeled data points for modern cloud-based cyber-attacks which make it ideal for experimentation in this study.

Three datasets are tested and examined in Section 4. The first dataset is a mechatronic simulation of a boom arm. In this experiment, the control signal is modified to modulate the position of the end effector of the boom arm. The pressure and force signals of the hydraulics in the experiment are examined and used to demonstrate the detector is robust to changes in scale.

The second dataset is a Power Electronic Converter (PEC) dataset where different fault conditions are introduced during normal operation. The goal of this experiment is to detect anomalies that fall outside the normal operating conditions or pattern of the converter. The faults in this experiment have significantly different characteristics and scale which demonstrate the detectors versatility.

The third dataset is a cyber-security dataset where attacks are identified in a target system following normal system operation. The goal of this experiment is to detect cyber-attacks and intrusions in computer infrastructure. This dataset is the largest and most complex presented in the study and the detector performs well on it.

Section 5 presents an analysis of the experimental datasets. Additionally the shortcomings of existing datasets and open research questions are discussed. The developed anomaly detector performed well on the three experimental datasets. The system accurately detected each change points in the small scale mechatronic simulation test. The detector identified 100% of the start and ends of the 4 faults in the PEC dataset. In the BETH dataset, the detector had some false positive defections but was able to identify 100% of the intrusions classified as dangerous.

Overall the proposed detection methodology provided a high rate of accuracy and precision in the experimental test cases. The detector can be further developed and improved to provide improved detection in increasingly challenging circumstances.

This work presents preliminary results demonstrating the applicability and performance of the proposed anomaly detection system. In the future, the detector can be developed further and incorporated into a production system for real-time monitoring and decision making to improve outcomes in a variety of interdisciplinary applications.

References

Alves, H., Sant'Ana, J., Rojas, D.G., Nardelli, P., and Pavol Mulinka, C.K., 2022. Demonstration descriptions. *Framework for the identification of rare events via machine learning and iot networks* [Online]. Available from: `https://fireman-project.eu/attachments/article/25/FIREMAN_Deliverable_D6_3.pdf`.

Andersson, H., Mariegaard, J.S., and Falk, A.K., 2021. *Tsod: anomaly detection for time series data* (v.0.1.3). Available from: `https://github.com/DHI/tsod` [Accessed November 7, 2021].

Beattie, A., 2021. *Adding local outlier factor (lof) algorithms*. GitHub. Available from: `https://github.com/online-ml/river/discussions/706`.

Beattie, A., 2022. *Master's thesis*. GitHub. Available from: `https://github.com/alexbeattie42/masters-thesis`.

Bhatnagar, A., Kassianik, P., Liu, C., Lan, T., Yang, W., Cassius, R., Sahoo, D., Arpit, D., Subramanian, S., Woo, G., Saha, A., Jagota, A.K., Gopalakrishnan, G., Singh, M., Krithika, K.C., Maddineni, S., Cho, D., Zong, B., Zhou, Y., Xiong, C., Savarese, S., Hoi, S., and Wang, H., 2021. Merlion: a machine learning library for time series. arXiv: `2109.09265 [cs.LG]`.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D., 2015. *Weight uncertainty in neural networks* [Online]. arXiv. Available from: `https://doi.org/10.48550/ARXIV.1505.05424`.

Castro Tomé, M.d., Gutierrez-Rojas, D., Nardelli, P.H.J., Kalalas, C., Silva, L.C.P.d., and Pouttu, A., 2022. Event-driven data acquisition for electricity metering: a tutorial. *Ieee sensors journal* [Online], 22(6), pp.5495–5503. Available from: `https://doi.org/10.1109/JSEN.2022.3147016`.

Chakraborty, S., Shah, S., Soltani, K., Swigart, A., Yang, L., and Buckingham, K., 2020. Building an automated and self-aware anomaly detection system. *Corr* [Online], abs/2011.05047. arXiv: 2011.05047. Available from: https://arxiv.org/abs/2011.05047.

Christou, I.T., Gutierrez-Rojas, D., Nardelli, P., Mulinka, P., Dzaferagic, M., and Macaluso, I., 2020. Initial results on detection and predicition techniques. *Framework for the identification of rare events via machine learning and iot networks* [Online]. Available from: https://fireman-project.eu/attachments/article/24/FIREMAN_Deliverable_D5_1__Copy_.pdf.

Dawy, Z., Saad, W., Ghosh, A., Andrews, J.G., and Yaacoub, E., 2017. Toward massive machine type cellular communications. *Ieee wireless communications* [Online], 24(1), pp.120–128. Available from: https://doi.org/10.1109/MWC.2016.1500284WC.

Fireman Project EU, 2021. *Fireman homepage* [Online]. Available from: https://fireman-project.eu [Accessed March 3, 2021].

Gal, Y. and Ghahramani, Z., 2016. Dropout as a bayesian approximation: representing model uncertainty in deep learning. *Proceedings of the 33rd international conference on international conference on machine learning - volume 48*, ICML'16. New York, NY, USA: JMLR.org, pp.1050–1059.

Gharghabi, S., Ding, Y., Yeh, C.-C.M., Kamgar, K., Ulanova, L., and Keogh, E., 2017. Matrix profile viii: domain agnostic online semantic segmentation at superhuman performance levels. *2017 ieee international conference on data mining (icdm)* [Online], pp.117–126. Available from: https://doi.org/10.1109/ICDM.2017.21.

Goodfellow, I.J., Shlens, J., and Szegedy, C., 2014. *Explaining and harnessing adversarial examples* [Online]. arXiv. Available from: https://doi.org/10.48550/ARXIV.1412.6572.

Gutierrez-Rojas, D., Christou, I.T., Dantas, D., Narayanan, A., Nardelli, P.H.J., and Yang, Y., 2022. Performance evaluation of machine learning for fault selection in power transmission lines. *Knowledge and information systems* [Online], 64(3), pp.859–883. Available from: https://doi.org/10.1007/s10115-022-01657-w.

Herzen, J., Lässig, F., Piazzetta, S.G., Neuer, T., Tafti, L., Raille, G., Pottelbergh, T.V., Pasieka, M., Skrodzki, A., Huguenin, N., Dumonal, M., Kościsz, J., Bader, D., Gusset, F., Benheddi, M., Williamson, C., Kosinski, M., Petrik, M., and Grosch, G., 2021. *Darts: user-friendly modern machine learning for time series* [Online]. arXiv: 2110.03224 [cs.LG].

Highnam, K., Arulkumaran, K., Hanif, Z., and Jennings, N., 2021. *Beth dataset: real cybersecurity data for anomaly detection research* [Online]. Unpublished. Available from: http://www.gatsby.ucl.ac.uk/~balaji/udl2021/accepted-papers/UDL2021-paper-033.pdf.

Kim, T. and Park, C.H., 2020. Anomaly pattern detection for streaming data. *Expert systems with applications* [Online], 149, p.113252. Available from: https://doi.org/https://doi.org/10.1016/j.eswa.2020.113252.

Lai, K.-H., Zha, D., Wang, G., Xu, J., Zhao, Y., Kumar, D., Chen, Y., Zumkhawaka, P., Wan, M., Martinez, D., and Hu, X., 2021a. Tods: an automated time series outlier detection system. *Proceedings of the aaai conference on artificial intelligence*, 35(18), pp.16060–16062.

Lai, K.-H., Zha, D., Xu, J., Zhao, Y., Wang, G., and Hu, X., 2021b. Revisiting time series outlier detection: definitions and benchmarks. *Thirty-fifth conference on neural information processing systems datasets and benchmarks track (round 1)* [Online]. Available from: https://openreview.net/forum?id=r8IvOsnHchr.

Law, S.M., 2019. STUMPY: A Powerful and Scalable Python Library for Time Series Data Mining. *The Journal of Open Source Software*, 4(39), p.1504.

Li, S., Xu, L.D., and Wang, X., 2013. Compressed sensing signal and data acquisition in wireless sensor networks and internet of things. *Ieee transactions on industrial informatics* [Online], 9(4), pp.2177–2186. Available from: https://doi.org/10.1109/TII.2012.2189222.

Lundberg, S. and Lee, S.-I., 2017. *A unified approach to interpreting model predictions* [Online]. arXiv. Available from: https://doi.org/10.48550/ARXIV.1705.07874.

Marrs, T., 2019. *Introduction to matrix profiles* [Online]. Available from: https://towardsdatascience.com/introduction-to-matrix-profiles-5568f3375d90 [Accessed March 4, 2022].

Medico, R., 2020. Rob-med/awesome-ts-anomaly-detection [Online]. Available from: https://doi.org/10.5281/zenodo.3972944.

Montiel, J., Halford, M., Mastelini, S.M., Bolmier, G., Sourty, R., Vaysse, R., Zouitine, A., Gomes, H.M., Read, J., Abdessalem, T., and Bifet, A., 2020. *River: machine learning for streaming data in python* [Online]. arXiv: 2012.04740 [cs.LG].

Na, G.S., Kim, D., and Yu, H., 2018. Dilof: effective and memory efficient local outlier detection in data streams. *Proceedings of the 24th acm sigkdd international conference on knowledge discovery & data mining* [Online], KDD '18. London, United Kingdom: Association for Computing Machinery, pp.1993–2002. Available from: https://doi.org/10.1145/3219819.3220022.

Rojas, D.G., Nardelli, P., Kalalas, C., Christou, I., and Papadias, C., 2020a. Design and functional architecture for data acquisition. *Framework for the identification of rare events via machine learning and iot networks* [Online]. Available from: https://fireman-project.eu/attachments/article/18/FIREMAN_Deliverable_D3_2%20(4).pdf.

Rojas, D.G., Nardelli, P., Kalalas, C., and Papadias, C., 2020b. Report on physical process data modeling. *Framework for the identification of rare events via machine learning and iot networks* [Online]. Available from: https://fireman-project.eu/attachments/article/18/FIREMAN_Deliverable_D3_1%20(5).pdf.

Rokrok, E., Qoria, T., Bruyere, A., Francois, B., and Guillaud, X., 2022. Transient stability assessment and enhancement of grid-forming converters embedding current reference saturation as current limiting strategy. *Ieee transactions on power systems* [Online], 37(2), pp.1519–1531. Available from: https://doi.org/10.1109/TPWRS.2021.3107959.

Sahoo, S., Wang, H., and Blaabjerg, F., 2021. On the explainability of black box data-driven controllers for power electronic converters. *2021 ieee energy conversion congress and exposition (ecce)* [Online], pp.1366–1372. Available from: https://doi.org/10.1109/ECCE47101.2021.9595231.

Salehi, M., Leckie, C., Bezdek, J.C., Vaithianathan, T., and Zhang, X., 2017. Fast memory efficient local outlier detection in data streams (extended abstract). *2017 ieee 33rd international conference on data engineering (icde)* [Online], pp.51–52. Available from: https://doi.org/10.1109/ICDE.2017.32.

Senin, P., Lin, J., Wang, X., Oates, T., Gandhi, S., Boedihardjo, A.P., Chen, C., and Frankenstein, S., 2018. Grammarviz 3.0: interactive discovery of variable-length time series patterns. *Acm trans. knowl. discov. data* [Online], 12(1), 10:1–10:28. Available from: https://doi.org/10.1145/3051126.

Souza Sant'Ana, J.M. de, Rodriguez, N.M., Eldeeb, E., Alves, H., Ullah, M., Nardelli, P.J., Kalalas, C., and Dzaferagic, M., 2020. Initial results on heterogeneous big data aggregation. *Framework for the identification of rare events via machine learning and iot networks* [Online]. Available from: https://fireman-project.eu/attachments/article/23/FIREMAN_Deliverable_D4_1%20(1).pdf.

Souza Sant'Ana, J.M. de, Eldeeb, E., Alves, H., Mulinka, P., Kalalas, C., Gutierrez-Rojas, D., and Nardelli, P.J., 2021. Plan the demonstrations and targets. *Framework for the identification of rare events via machine learning and iot networks* [Online]. Available from: https://fireman-project.eu/attachments/article/9/FIREMAN_Deliverable_D6_1.pdf.

Tan, S.C., Ting, K.M., and Liu, T.F., 2011. Fast anomaly detection for streaming data. *Proceedings of the twenty-second international joint conference on artificial intelligence - volume two*, IJCAI'11. Barcelona, Catalonia, Spain: AAAI Press, pp.1511–1516.

Togbe, M.U., Chabchoub, Y., Boly, A., Barry, M., Chiky, R., and Bahri, M., 2021. Anomalies detection using isolation in concept-drifting data streams. *Computers* [Online], 10(1). Available from: `https://doi.org/10.3390/computers10010013`.

Tsuruta, H. and Feng, G., 2021. *Banpei* (v.0.1.2). Available from: `https://github.com/tsurubee/banpei`.

UCI Machine Learning Repository, 1998a. *Covertype data set* [Online]. Available from: `https://archive.ics.uci.edu/ml/datasets/covertype`.

UCI Machine Learning Repository, 1998b. *Pen-based recognition of handwritten digits data set* [Online]. Available from: `https://archive.ics.uci.edu/ml/datasets/Pen-Based+Recognition+of+Handwritten+Digits`.

UCI Machine Learning Repository, n.d.(a). *Japanese vowels data set* [Online]. Available from: `https://archive.ics.uci.edu/ml/datasets/Japanese+Vowels`.

UCI Machine Learning Repository, n.d.(b). *Statlog (shuttle) data set* [Online]. Available from: `https://archive.ics.uci.edu/ml/datasets/Statlog+(Shuttle)`.

Ullah, M., Nardelli, P.H.J., Wolff, A., and Smolander, K., 2020. Twenty-one key factors to choose an iot platform: theoretical framework and its applications. *Ieee internet of things journal* [Online], 7(10), pp.10111–10119. Available from: `https://doi.org/10.1109/JIOT.2020.3000056`.

University of California, Irvine, 1999. *Kdd cup 1999 data* [Online]. Available from: `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`.

Van Looveren, A., Vacanti, G., Klaise, J., Coca, A., and Cobb, O., 2019. *Alibi detect: algorithms for outlier, adversarial and drift detection* (v.0.7.3). Available from: `https://github.com/SeldonIO/alibi-detect` [Accessed October 29, 2021].

Yeh, C.-C.M., Zhu, Y., Ulanova, L., Begum, N., Ding, Y., Dau, A., Silva, D., Mueen, A., and Keogh, E., 2016. Matrix profile i: all pairs similarity joins for time series: a unifying view that includes motifs, discords and shapelets [Online], pp.1317–1322. Available from: https://doi.org/10.1109/ICDM.2016.0179.

Yilmaz, S.F. and Kozat, S.S., 2020. Pysad: A streaming anomaly detection framework in python. *Corr* [Online], abs/2009.02572. arXiv: 2009.02572. Available from: https://arxiv.org/abs/2009.02572.

Yu, K., Shi, W., and Santoro, N., 2020. Designing a streaming algorithm for outlier detection in data mining—an incrementa approach. *Sensors* [Online], 20(5). Available from: https://doi.org/10.3390/s20051261.

Zhao, Y., Nasrullah, Z., and Li, Z., 2019. Pyod: a python toolbox for scalable outlier detection. *Journal of machine learning research* [Online], 20(96), pp.1–7. Available from: http://jmlr.org/papers/v20/19-011.html.

Zhu, Y., Imamura, M., Nikovski, D., and Keogh, E., 2017. Matrix profile vii: time series chains: a new primitive for time series data mining (best student paper award). *2017 ieee international conference on data mining (icdm)* [Online], pp.695–704. Available from: https://doi.org/10.1109/ICDM.2017.79.